ISSN 1213-6972 *Volume 27, Number 2, 2019*

Journal of WSCG

An international journal of algorithms, data structures and techniques for computer graphics and visualization, surface meshing and modeling, global illumination, computer vision, image processing and pattern recognition, computational geometry, visual human interaction and virtual reality, animation, multimedia systems and applications in parallel, distributed and mobile environment.

Vaclav Skala – Union Agency

Journal of WSCG

Editor-in-Chief: Vaclav Skala
c/o University of West Bohemia
Faculty of Applied Sciences
Univerzitni 8
CZ 306 14 Plzen
Czech Republic
http://www.VaclavSkala.eu

Managing Editor: Vaclav Skala

Printed and Published by:
Vaclav Skala - Union Agency
Na Mazinach 9
CZ 322 00 Plzen
Czech Republic

Published in cooperation with the University of West Bohemia
Univerzitní 8, 306 14 Pilsen, Czech Republic

Hardcopy: ***ISSN 1213 – 6972***
CD ROM: ***ISSN 1213 – 6980***
On-line: ***ISSN 1213 – 6964***

Journal of WSCG

Editor-in-Chief

Vaclav Skala

c/o University of West Bohemia
Faculty of Applied Sciences
Department of Computer Science and Engineering
Univerzitni 8, CZ 306 14 Plzen, Czech Republic
http://www.VaclavSkala.eu

Journal of WSCG URLs: http://www.wscg.eu or http://wscg.zcu.cz/jwscg

Editorial Board

Journal of WSCG

Board of Reviewers
2019

Anderson,M. (Brazil)
Assarsson,U. (Sweden)
Ayala,D. (Spain)
Baranoski,G. (Canada)
Benes,B. (USA)
Benger,W. (Austria)
Bilbao,J. (Spain)
Birra,F. (Portugal)
Bouatouch,K. (France)
Boukaz,S. (France)
Bourke,P. (Australia)
Cakmak,H. (Germany)
Carmo,M. (Portugal)
Carvalho,M. (Brazil)
Coquillart,S. (France)
Dachsbacher,C. (USA)
De Martino,J. (Brazil)
Durikovic,R. (Slovakia)
Eisemann,M. (Germany)
Feito,F. (Spain)
Ferguson,S. (United Kingdom)
Galo,M. (Brazil)
Gavrilova,M. (Canada)
Gdawiec,K. (Poland)
Gerhards,J. (France)
Giannini,F. (Italy)
Goncalves,A. (Portugal)
Gudukbay,U. (Turkey)
Guthe,M. (Germany)
Hansford,D. (USA)
Hermosilla Casajus,P. (Germany)
Chaudhuri,D. (India)
Chover,M. (Spain)
Ivrissimtzis,I. (United Kingdom)
Jeschke,S. (Austria)
Joan-Arinyo,R. (Andorra)
Jones,M. (United Kingdom)
Juan,M. (Spain)
Kanai,T. (Japan)
Klosowski,J. (USA)
Komorowski,J. (Poland)
Krueger,J. (Germany)
Kurt,M. (Turkey)
Kyratzi,S. (Greece)
Larboulette,C. (France)
Lee,J. (USA)

Lisowska,A. (Poland)
Liu,S. (China)
Lobachev,O. (Germany)
Manoharan,P. (India)
Manzke,M. (Ireland)
Mastermayer,A. (Germany)
Max,N. (USA)
Meyer,A. (France)
Mohd Suaib,N. (Malaysia)
Molla,R. (Spain)
Montrucchio,B. (Italy)
Muller,H. (Germany)
Nishio,K. (Japan)
Oliveira,J. (Portugal)
Oyarzun Laura,C. (Germany)
Pan,R. (China)
Papaioannou,G. (Greece)
Paquette,E. (Canada)
Pasko,A. (United Kingdom)
Pedrini,H. (Brazil)
Ramires Fernandes,A. (Portugal)
Richardson,J. (USA)
Rodrigues,N. (Portugal)
Rojas-Sola,J. (Spain)
Sarfraz,M. (Kuwait)
Savchenko,V. (Japan)
Segura,R. (Spain)
Shum,H. (United Kingdom)
Sik-Lanyi,C. (Hungary)
Skala,V. (Czech Republic)
Sousa,A. (Portugal)
Stuerzlinger,W. (Canada)
Szecsi,L. (Hungary)
Thalmann,D. (Switzerland)
Todt,E. (Brazil)
Tokuta,A. (USA)
Toler-Franklin,C. (USA)
Torrens,F. (Spain)
Trapp,M. (Germany)
Tytkowski,K. (Poland)
Vanderhaeghe,D. (France)
Weber,A. (Germany)
Wuensche,B. (New Zealand)
Wuethrich,C. (Germany)
Cai,Y. (Singapore)
Yin,Y. (USA)
Yoshizawa,S. (Japan)
Zwettler,G. (Austria)

Journal of WSCG

Vol.27, No.2, 2019

Contents

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Lens-based Focus+Context Visualization Techniques for Interactive Exploration of Web-based Reachability Maps

Marc Listemann
Institute of Data Science
German Aerospace Center (DLR)
Germany
marc.listemann@dlr.de

Matthias Trapp
Hasso Plattner Institute
Faculty of Digital Engineering
University of Potsdam, Germany
trapp@hpi.de

Jürgen Döllner
Hasso Plattner Institute
Faculty of Digital Engineering
University of Potsdam, Germany
doellner@hpi.de

ABSTRACT

Reachability maps are powerful means to help making location-based decisions, such as choosing convenient sites for subsidiaries or planning vacation trips. Existing visualization approaches, however, introduce drawbacks concerning an effective acquisition of relevant information and an efficient data processing. In this paper, we introduce the first approach known so far to apply focus+context techniques to web-based reachability maps. We therefore propose a real-time enabled combination of an isochrone-based and a network-based representation of travel times obtained from multi-modal routing analysis using interactive lenses. We furthermore present a GPU-accelerated image-based method to compute isochrones based on a travel time-attributed network on client-side and thus achieve reduced data transmission efforts while yielding isochrones of higher precision, compared to generalized geometry-based approaches.

Keywords

focus+context, geovisualization, web maps, mobility analytics, real-time rendering, interactive lenses

1 INTRODUCTION

Reachability analyses aim to inform about the potential of an object to reach all locations in a given area from a starting point. To this effect, the "Single-source Shortest-path (SSSP)" problem has to be solved, as it finds all shortest paths between a starting point V_s and every possible node V_i in a graph-based network. The result are computed travel times for every node, representing the potential of an object using a specific transport medium at V_s to reach the respective nodes. As a result, several conceivable applications arise from reachability analytics, ranging from leisure time activity planning to complex site analysis [32].

1.1 Problem Statement

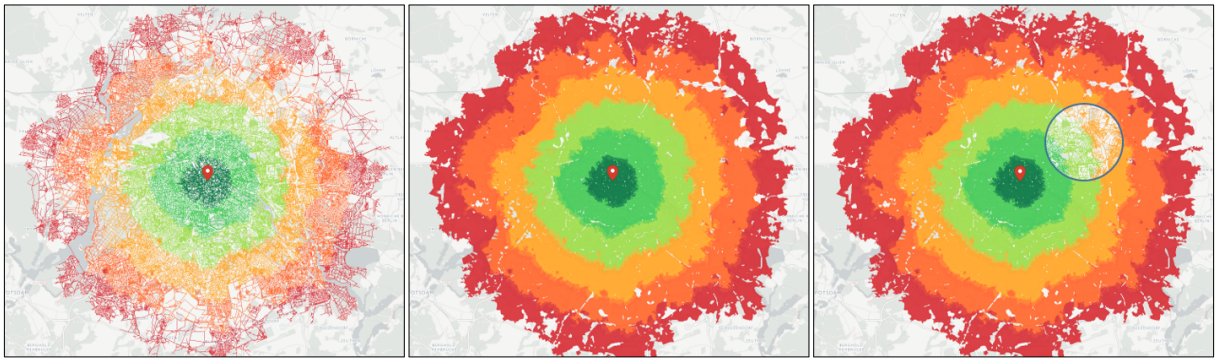
The computed travel times can consequently be visualized on a map where they are usually grouped into intervals by assigning them colors. Modern technologies allow for a web-based representation of reachability maps, thus providing a fast and interactive access to travel time information, whereas expensive computations are performed in the backend. With respect to 2D web-map applications, we emphasize two basic approaches for the visualization of reachability data: *Isochrone-based Visualization (IBV)* and *Network-based Visualization (NBV)*.

In this paper, we refer to isochrones as generalized areas of equal travel times, being composed into intervals by assigning meaningful colors (Fig. 1b). The NBV, on the other hand, denotes a direct color mapping of travel time information onto the edges of the geographic network, as proposed by Schoedon *et al*. [23] (Fig. 1a). It therefore offers a high degree of detail, since the rendered lines directly refer to the streets in the network and no pre-generalization is applied, as it is the case for many isochrone computing procedures.

However, both approaches introduce implicit challenges regarding the user's potential to effectively obtain relevant information from the map:

C1: Isochronal representations lack a detailed view on the network due to generalization measures. Therefore, they may cause interpretation errors in areas of interest.

C2: Network-based representations impede a fast survey of the overall situation in the specified area concerning travel time information. They are likely to introduce interpretation difficulties and visual clutter at low zoom levels.



(a) Network-based visualization. (b) Isochrone-based visualization. (c) Lens-based visualization.

Figure 1: The basic concept of the lens-based focus+context visualization technique proposed in this paper. Two generated representations with different information density for focus and context, i.e., a network-based (a) and an isochrone-based representation (b) are combined to a single visualization using a magic lens metaphor (c).

C3: Conventional methods for generating isochrones rely on computation of geometry data and thus implicate a massive data transfer between server and client which involves latencies.

Hence, a concept is required that (1) combines the advantages of IBVs and NBVs (i.e., an overview-giving impression and a detailed view of reachability information) in a single visualization and (2) establishes a method for the generation of isochrones that addresses C3 and takes advantage of the properties of the Graphics Processing Unit (GPU), allowing for real-time rendering of massive data sets [30, 28]. The overall goal of this paper is therefore to support the user of a reachability map to effectively extract relevant information by retaining overview simultaneously while being provided with real-time enabled interaction methods.

1.2 Approach and Contributions

To this effect, *focus+context* is a powerful paradigm to improve the user's ability to extract relevant information out of a visualization as it represents detailed information in *focus* areas, which are embedded and linked to a *context* view, thus taking advantage of the available viewport space [6]. This work makes use of a prevalent Focus+Context Visualization (FCV) that is known as *Magic Lens* [27] and combines the two mentioned approaches to gain a user-defined focal view inside the surrounding isochrones (Fig. 1c), ensuring an appropriate and real-time visualization as well as a high degree of interaction. It is prototypical implemented as a web-map application using state-of-the-art web technologies, e.g., Web Graphics Library (WebGL) and HTML5 to obtain GPU-accelerated run-time performance.

For it, the GL Transmission Format (glTF) tiling approach by Schoedon *et al.* is being extended, while benefiting from GPU-enabled buffers containing the geographic network and the computed travel times [23]. Originally used for the rendering of travel time-attributed lines to obtain a detailed representation of reachability, it can be shown that this technique represents the fundamentals for the generation of an overview or context representation based on isochrones, too. For the isochrone generation, a GPU-based image-processing technique called Jump-Flooding Algorithm (JFA) [21] is adapted that allows for efficient client-side rendering. To summarize, this paper makes the following contributions:

1. It presents a concept and prototypical implementation of a lens-based FCV for reachability maps in the context of mobility analytics.

2. It describes an approach to generate isochrones using the JFA that is capable of real-time performance in a browser-based visualization environment.

The remainder of this paper is structured as follows. Sec. 2 reviews related work on interactive reachability maps and lens-based focus+context techniques. Sec. 3 introduces the concept of our approach and provides details for the proposed web-based isochrone generation method. Sec. 4 evaluates the results of that method, followed by a discussion of its run-time rendering performance as well as conceptual and technical limitations, and states possible future research directions. Finally, Sec. 5 concludes this paper.

2 RELATED WORK

The related work is surveyed with respect to previous approaches on travel time visualization and FCV techniques for 2D geographic data.

2.1 Reachability Maps

Previous work on reachability maps uses *IBV* [9, 3] or *NBV* [12, 23], whereas the latter does not scale for low

zoom levels. In the work of Spiekermann, transportation networks are distorted into so-called *time-space maps* whose scale does not rely on spatial metrics but on travel time units [25]. Such representation is used by Lemke *et al*. to demonstrate shrinking regional disparities inside Europe in an economical context [18]. Whereas the time-space map approach may be supportive in this specific use case, it lacks obtaining concrete travel time values for a certain region. Besides that, an adaption to large scale applications such as web maps would not be sufficient as the map implicitly distorts outwards and small-scale distortions are hardly noticeable. Moreover, the authors indicate that the visualization may cause interpretation mistakes, as regions between the calibration points, which are considered major cities, could lose their accessibility, which is not represented in the map, in fact.

On the other hand, the work of Carden proposes an alternative displaying of reachability information within a map [5]. Similar to the tube map of London, a web-based application for the interactive reachability analysis was developed. Therefore, travel times from a selected subway station are computed and visualized by laying circular isochrones around the respective station and distorts the tube map in order to fit into the isochrones. A similar approach is used to visualize travel time data for cities in the Netherlands [19]. Instead of distorting a public transportation network, the map itself is distorted to fit the circular isochrones.

2.2 Focus+Context for Map Visualization

Focus+context research dates back to the 1980s, where Furnas developed "Generalized Fisheye Views" [8]. This distortion-based lens metaphors have been refined by various research [11]. Meanwhile, a number of different focus+context techniques have been established, though a generally accepted systematization is still missing [16, 6, 4]. However, Lokuge and Ishizaki applied these techniques to geographic data by developing an interactive map system that reacts on user queries [13]. It manipulates typography, transparency and color to alter the appearance of geographical elements due to the interest of the user. In this context, *Visugrams* enable simultaneous highlighting of relevant geographic information and related abstract information selected by the user [7].

Concerning focus+context applications for transportation networks, research has been examined for navigational problems. As such, Agrawala and Stolte present improved static route maps derived from generalization techniques [1]. They analyze hand-drawn route maps as well as cognitive aspects in psychology, to determine the characteristics of an effective route map, that hides dispensable streets and emphasizes significant route elements. A real-time enabled system for automatic generation of enhanced route maps is further being developed, performing sophisticated generalization steps. An improved method to this approach is furthermore introduced by Kopf *et al*. [15]. Karnick *et al*. , on the other hand, establish a multi-focal technique that generates detail lenses related to specific Point-of-Interests (POIs) along the route [14]. These should alleviate the readability of the map by magnifying those parts of the route that require an action by the driver. However, although their approach is carried out in a web-based automated system, it is developed for non-interactive, static maps too.

Instead of applying the frequently used fisheye distortion, Haunert and Sering suggest for a graph-based optimization approach in road networks to properly display a focal region while distorting only the sparse parts of the network [10]. Based on a user-defined Region-of-Interest (ROI) and zoom factor, the focus region is scaled up by that factor and the context is scaled down while minimizing the square sum of distortions. Though being able to run in polynomial time, the approach lacks real-time capability.

Krause *et al*. introduce a displacement technique for a public transportation system in Konstanz, Germany. They aim to visualize travel times between different network elements and to facilitate comparisons between different bus routes [17]. This is implemented by mapping computed travel times to visual distances of the edges. They develop i.a. a radial layout for representing the results of a routing query. It places bus stations concentric around a starting position and preserves topological characteristics by aligning the edges between the stations according to their geographic positions. However, this approach is can hardly be adapted to general transportation systems.

However, a real-time capable focus+context technique proposed by Wang and Chi, focus on the visualization of routes in public transport networks, e.g., metro maps [31]. By highlighting those parts of the subway network that are members of the best route to a destination and providing more space the related stations, they enable an adequate visualization of complex metro maps on small display areas. Additionally, to facilitate the map reading, they apply octilinear transportation lines and equal distances between the stations. The final layout is adapted due to the least square solving.

Furthermore, techniques such as semantic depth-of-field [16] can be applied to highlight geographic line elements on a map [29]. These allow for an assignment of relevance indicating values to the data and thereby carrying out the decision of an element being considered part of the focus or the context.

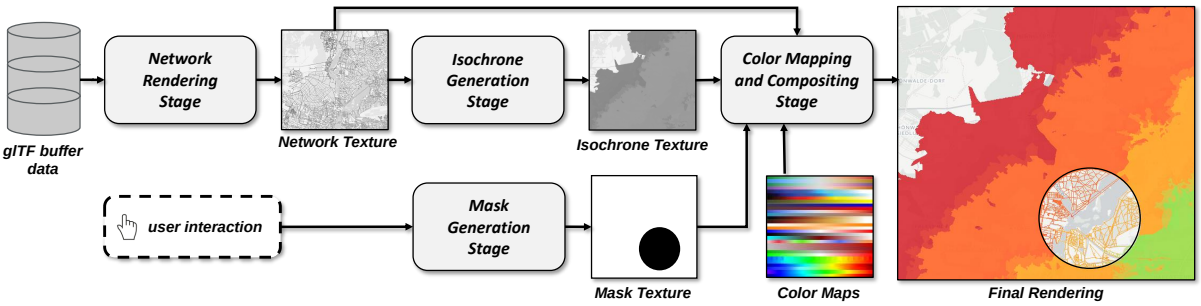


Figure 2: Conceptual overview of the presented approach to generate a FCV for reachability maps. Only the frontend is illustrated, as it is the focus of this work. Several rendering passes are executed to create three textures that are merged in a compositing pass.

3 VISUALIZATION APPROACH

This section presents the concept of implementing a FCV by combining IBV and NBV maps. Based on requirements (Sec. 3.1), an overview (Sec. 3.2) and implementation details are described (Secs. 3.3 to 3.5).

3.1 Requirements and Preliminaries

According to the challenges related to existing visualization approaches using reachability maps (Sec. 1), requirements arise for a method to deal with these. A FCV technique for web-based reachability maps should therefore provide:

R1: an appropriate visualization to enable the user to view both, a detailed and a compact representation of reachability data within one display. This addresses challenges C1 and C3 to facilitate a more effective visual analysis of the geographic network. Accordingly, the visualization should provide users with a detailed view on relevant information without losing track of the context.

R2: an application responding on user interaction in real-time, addressing the need for interactivity of focus+context applications [28]. Since the user should be enabled to define a focal region by themselves, they will expect an instant visual feedback.

R3: an alternative method to server-based Geometry-based Isochrones (GBI) generation that reduces both, data transmission and processing effort. This addresses C2 and defines the need to develop an alternative method to the data-intensive conventional methods for isochrone generation. It is related to R2 as the isochrone generation method should be able to produce instant visual results according to modifications of user-defined parameters (e.g., the starting position or travel time ranges).

R4: a graphical user interface that assist users in the decision making process. This is important, as the proposed method does not primarily intend to predict ROI for the user, which is not the task of a reachability map. In fact, it aims to assist the user discovering a ROI. This can be achieved by allowing to arbitrarily define stylization parameters and other variables of the visual output. It is related to R2, since the real-time responses are essential for interactivity.

For these purposes, the Magic Lens metaphor stands out among the FCV techniques, as it enables users to interactively observe different visual appearances of the data while having full control over the ROI. It provides the most detailed representation of reachability information (i.e., the NBV) inside its outlines while keeping a generalized view (i.e., the IBV) surrounding. R2 and R3 will be addressed by applying sophisticated GPU-accelerated techniques on the client-side.

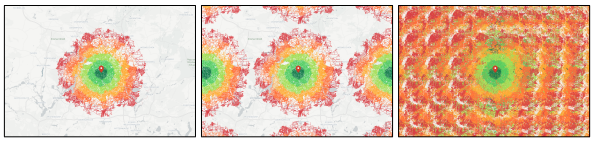
3.2 Conceptual Overview

Our approach extends the work of Schoedon *et al.* [23] and therefore builds on backend-provided glTF buffers, containing the street network and the respective travel times. After their initial rasterization, we are able to create the context representation out of the focus representation while making use of a sophisticated information distribution algorithm. The particular components of the FCV (i.e., focus, context, and ROI mask) will eventually be composed at the end of the rendering process. Fig. 2 shows a conceptual overview comprising the control and data flow between the following stages described in the remainder of this section:

Network-Rendering Stage: rasterizes the network geometry into a G-Buffer representation [22]. This step follows Schoedon et al. [23] and will therefore not be covered in this paper.

Isochrone-Generation Stage: comprises an initialization pass for generating an initial texture as input for subsequent flooding passes that distribute travel times over the map to obtain isochrones (Sec. 3.3).

Mask-Generation Stage: generates a mask texture representing the ROI (magic lenses or boundaries) and is triggered by user interaction (Sec. 3.4).



(a) (b) (c)

Figure 3: Visualization (including color mapping) of JFA steps for an input network (a): two rounds (b) and four rounds (c).

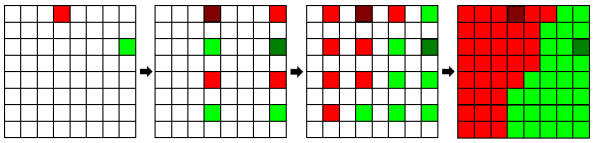


Figure 4: Working principle of JFA exemplified on an 8×8 grid and two seed points. Three rounds are required to fill the image.

Color Mapping and Compositing Stage: uses the results of the network rendering pass, the flooding passes, and the mask-generation pass to create the respective mapping from travel time to color, and combines them into a single image (Sec. 3.5).

3.3 Image-based Isochrone Generation

Our method for the generation of Image-based Isochrones (IBI) is based upon distributing data from set pixels to adjacent non-attributed pixels, resulting in *Voronoi cells*. For it, we make use of the Jump-Flooding Algorithm (JFA) [21] that allows for fast propagation of pixels' contents to their closest neighbors in constant time whereas being independent of the number of input pixel elements (i.e., *seed points*) and capable of running in parallel on the GPU. Instead of applying CPU-consuming computations of isochrone geometry, we are therefore able to obtain the context representation directly from the focus representation by making use of GPU-accelerated image processing techniques.

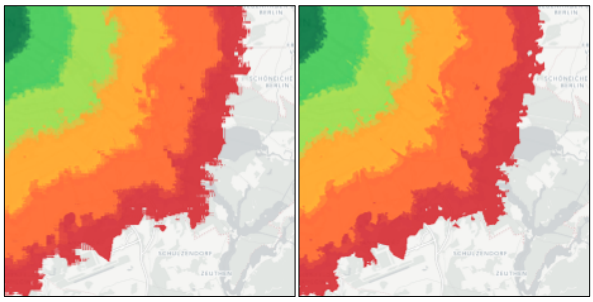
JFA-driven Propagation of Travel Time. The JFA performs a flooding of information over a target space: given a grid of size $w \times h$ (i.e., the input image) containing the seed points which values should be propagated, the algorithm ensures that each grid cell (i.e., a pixel) has correspondence to its closest seed. For it, the JFA distributes a pixel's information in a total of $\log_2(\max(w,h))$ rounds with varying step lengths, using the information of already passed pixels. Its basic functionality is illustrated in Fig. 4, whereas Fig. 3 depicts different jump flooding steps for a colored input street network.

To obtain Voronoi diagrams, the flooded pixels ultimately need to acquire the desired information (typically, the color) from their corresponding seed points. Therefore, Rong and Tan suggest for using the spread seed's coordinates of the jump flooding result to refer to an initially created texture for picking up the respective color values and dying the final image accordingly. However, this procedure requires an additional rendering pass that applies an initial color mapping to the seed points and, moreover, complicates a real-time enabled and interactive stylization of the isochrones with custom color ramps (Sec. 3.5) since all jump flooding passes need to be executed again whenever the style parameters are being changed. To avoid these issues, we adapt the original JFA approach by augmenting the information to be passed with another component. Alongside the seeds' positions we furthermore pass the travel time of the corresponding nodes of the transportation network by making use of an idle texture channel. In doing so, the color mapping can be carried out directly on the jump flooding result where every pixel of the viewport contains the travel time information of its closest supplying element and can be dyed according to that information.

WebGL Implementation Aspects. For the implementation of the JFA we make use of the WebGL Application Programming Interface (API) as it enables access to graphics hardware resources from a web browser using JavaScript (R2). Since our approach addresses image-based processing techniques and multiple rendering passes, the corresponding means of choice in WebGL are *textures*.

To ensure a proper functionality of the JFA, the following considerations about the textures' structure should be taken into account. WebGL 1.0 supports textures with four channels, i.e., Red-Green-Blue-α (RGBA). Since the individual channels store numeric values we can make use of them for image-based General-Purpose Computing on Graphics Processing Units (GPGPU) and store the x,y-coordinates of the seed points in the r,g-channels of the texture. The fragment shader's output variable in the OpenGL ES Shading Language (GLSL ES) 1.0 specification requires each element to range between 0.0 and 1.0. Thus, the seed's screen-space coordinates are normalized by dividing them by the resolution of the texture. Since our approach comprises furthermore the distribution of travel times, they are being stored in the alpha-component of the RGBA-texture. In addition, the blue-component will be used to store a Boolean-like value, indicating whether a pixel has already been visited by the flood or not.

By default, WebGL 1.0 assigns the `gl.UNSIGNED_BYTE` data type to texture objects which refers to 8 bit per channel and therefore 256 different values. The consequence is an internal mapping (clamp and quantization) of the $[0.0, 1.0]$ ranging floating point values to the $[0, 255]$ integral range which implies rounding and clamping operations. When retrieving the original



(a) 8-bit channel precision. (b) 32-bit channel precision.

Figure 5: Visual artifacts caused by precision issues, using a texture with unsigned byte precision (a) and with floating-point precision (b).

value through texture look-up, another internal remapping takes place, ensuring the texture values to be in $[0.0, 1.0]$ range again. Especially when dealing with high precise values, such as normalized screen-space coordinates of fragments, information is likely to get lost due to rounding issues, which could, in this case, result in visual artifacts caused by miscalculated seed positions, pointing to an incorrect pixel (Fig. 5). Therefore, we propose using a high-precision floating-point texture with a bit depth of 32 per channel, hence allowing for the storage of real numbers in the respective channels. This format is provided by the `OES_texture_float` extension and enables the definition of a `gl.FLOAT` texture, which prevents internal mappings of the normalized floating-point values.

Besides their main usage of determining a pixel's color on an object's surface based on images, textures can also be used as render targets in the form of G-Buffers [22] when they are attached as an additional color buffer to a custom Framebuffer Object (FBO). Since this technique enables rendering into a different target than the default framebuffer it is referred to as *off-screen rendering* which is a common tool in computer graphics for post-processing of previously rendered data to apply image-based effects. The geometry the off-screen texture is drawn to is considered a *screen-aligned quad*, since the application aims to display the processed texture on the whole canvas.

This concept of off-screen rendering and FBO-attached textures is indispensable for the implementation of the JFA, particularly because it is concerned an iterative multi-pass procedure where previously processed data needs to serve as input in the next iteration. To this effect, the principle of *ping-pong buffers* [2] allows for data processing between two off-screen textures, where the read-from-texture provides the input for the operation and the write-to-texture serves as render target for the processing results. After the execution of such a pass, or respectively, a JFA fragment shader, the two textures swap their roles. This procedure is repeated until a stop criterion has reached, in this case, a jump flooding step length of 1. The result of the JFA passes will be an off-screen texture, containing travel time values for every texel and accordingly being ready for color mapping (Sec. 3.5).

3.4 Mask Generation Stage

As mentioned in Sec. 1, we use the Magic Lens metaphor to combine the detailed NBV with the overview-giving IBV. In contrast to conventional lens approaches in information visualization [27], the lenses in the scope of our work do not literally execute a dedicated lens function, such as magnification or fish-eye distortion, that creates the focus representation on-the-fly.

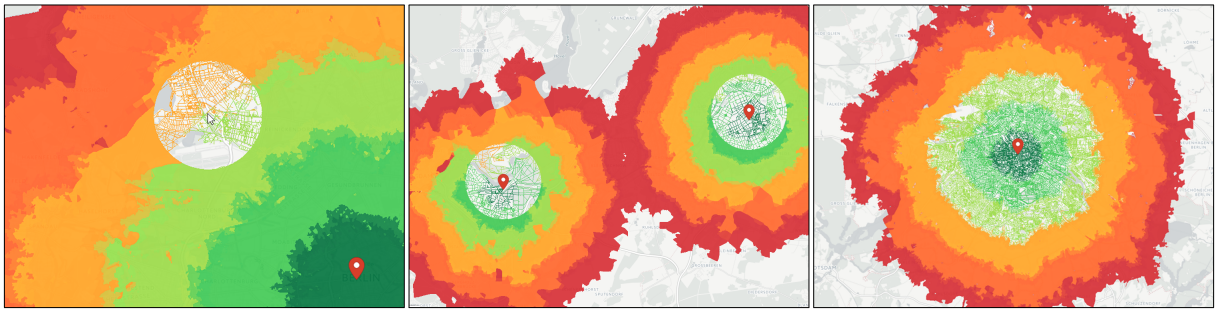
Instead, our lens purely serves as a mask, represented by a function of its position and shape, to determine where the previously created textures will be displayed. To meet R2 and R4, we conceive intuitive and interactive lenses, since the ROI is hardly predictable. As a result, most of our lenses follow the circular shape of the historically well-known magnifying glass and are highly user-defined. The following lens types are supported:

Interactive Lens: This user-controlled lens is interactively moveable across the map according to the mouse cursor (Fig. 6a). Since the user makes a decision according to unknown preferences, this lens suits well for reachability analytics. Its position is therefore specified by the mouse pointer's screen coordinates and the radius can be user-defined.

Marker Lens: Since reachability analysis allows for the definition of multiple starting points, e.g., in the scope of business location analytics, the availability of multiple lenses could be meaningful too. Hence, automatic lens placement at respective markers is presented (Fig. 6b). They are placed around the center of a position marker and therefore highlight the immediate surroundings of a POI. They can have a static radius but also be defined according to user-specific parameters such as distance from a main location or several statistical variables.

Isochronal Lens: The shape of this lens type directly corresponds to the travel times while being positioned at respective markers (Fig. 6c). Given a specific travel-time range defined by the user, the NBV appears within this range while displaying isochrones in the surrounding context. According to the definition of [26], this can be denoted as a smart lens, as it considers data-driven aspects of picking out values within a specific range (travel times).

Depending on the user's choice of an appropriate lens type, the respective shape will be rendered into the off-screen mask texture and hence be available for real-time enabled blending operations in the compositing stage.



(a) Interactive Lens. (b) Marker Lens. (c) Isochronal Lens.

Figure 6: Different types of lenses supported by the presented approach.

3.5 Color Mapping and Compositing

The result of the jump flooding passes is not yet the desired IBV but merely an image with travel time values in the α-component of every pixel. To achieve an impression of areal isochrones, the pixels on the map have to be colored on the basis of their travel time. Therefore, a dedicated shader has to be executed to perform a sampling of a custom color ramp texture based on the normalized travel times of the jump flooding texture and make the resulting context texture available in the memory.

Since the JFA propagates information over the complete canvas (Fig. 3c), we furthermore required to request a higher maximum travel time from the routing server and accordingly clamp the travel times to be displayed with a lower value in the fragment shader by discarding fragments with a travel time value above the threshold. This also allows for a smooth user-defined adjustment of travel times, e.g., by using a slider.

Provided the mask texture, representing the lens's shape (Sec. 3.4), a final compositing pass combines the previously generated focus and context textures using *alpha blending* [2]. In GLSL ES, the *over* operator is natively implemented by the `mix` function that actually performs a linear interpolation between the color vectors of the focus and context textures using the mask's α-value. By setting the mask's α-values to 1.0 where the lens's shape should be displayed and to 0.0 elsewhere, the over operator merely performs a conditional function to determine if either the focus texture or the context texture should be rendered.

Further variations of the lens presentation can be achieved by using a gradient mask texture with varying α-values for the lens's fragments [24]. By decreasing the α-value from the lens interior outwards, a smoother transition between focus regions and the context can be accomplished. This technique therefore represents a more strictly adaption to the focus+context definition of Cockburn [6] who stresses a seamless transition between focus and context to be required for FCVs.

4 RESULTS AND DISCUSSION

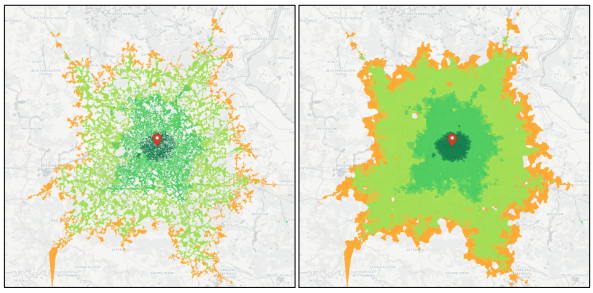
This section discusses the results of the proposed approach by evaluating our method for the generation of isochrones (Sec. 4.1), followed by a performance measurement using different web-browsers on mobile clients (Sec. 4.2), and a discussion of limitations as well as future research directions (Sec. 4.3).

4.1 Isochrone Evaluation

One main contribution of this work is the development of an alternative, image-based technique for generating isochrones of travel time data. It is hence worthwhile to examine the results of our approach and to compare them with established isochrone generation methods.

Accuracy Issues. Fig. 7 illustrates a noteworthy behavior of our IBI approach, associated with the JFA. Referring to [23], we allow for the selection of different transportation media (e.g., car, bike, public transit) in our web application. The JFA, however, will propagate also travel time information from street segments that are not accessible by the specified medium – a motorway, e.g., is forbidden for cyclists – and therefore causes remarkable gaps inside the isochrones due to subsequent discarding of fragments (Sec. 3.5). Although this behavior depicts a correct representation of the underlying data, it provokes a considerable amount of visual clutter and makes it difficult to distinguish any spatial relations in the disrupted isochrones (Fig. 7a). Since the task of the context in a FCV, though, is to provide an overview of the data, we suggest to carry out a pre-filtering of the travel time data by assigning those gap-causing fragments a specific value in the initialization pass, indicating they are not considered seed points and therefore not being distributed by the JFA. The resulting loss of accuracy, however, can be compensated by using our proposed Magic Lens as the detail information of the gaps will be shifted to the focus representation inside the lens.

Comparison of Approaches. A comparison of visualization results using IBI, as proposed in this paper, and the conventional approach of server-side generated GBI

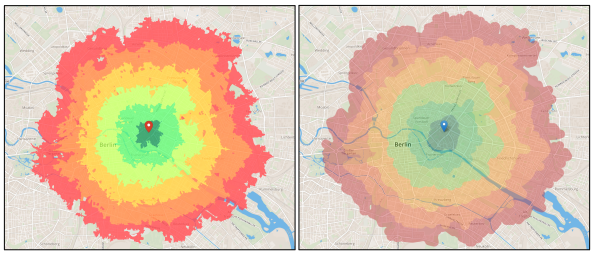


(a) Unfiltered travel time data, causing large amount of gaps. (b) Pre-filtering of travel time for flooding into invalid cells.

Figure 7: Comparison of results using unfiltered and pre-filtered travel time data.

is depicted in Fig. 8. Since the underlying algorithms of both approaches are completely different, a comparison can rather happen on the basis of visual results. Therefore, a single-marker application for both approaches is established that displays isochrones from 5 to 30 minutes of travel time in five-minutes steps. Although they appear rather similar, the GBI provides a smoother appearance than the IBI, which provides sharp-edged structures due to the nature of the JFA. Furthermore, it can be noticed that the IBI approach offers a considerable higher Level-of-Detail (LOD) whereas the GBI approach performs a server-side generalization that may result in inconsistencies regarding a correct representation of travel times, as some areas are depicted reachable though they are actually not.

However, the LOD of the GBI varies with different zoom levels, which can be considered a desired behavior. The IBI approach is yet not able to vary its LOD, though the aforementioned method of filling the gaps is an attempt to simulate a lower LOD. Due to the need for zooming into the map to survey detailed representations with the GBI approach, the user might lose track of the overall situation, whereas the IBI in combination with the Magic Lens allow for a combined visualization of detail and overview within one viewport. For that reason and since both approaches have a similar appearance whereas the IBI provides more detail, it can be considered an adequate alternative to the established GBI approaches.



(a) Image-based isochones. (b) Geometry-based isochones.

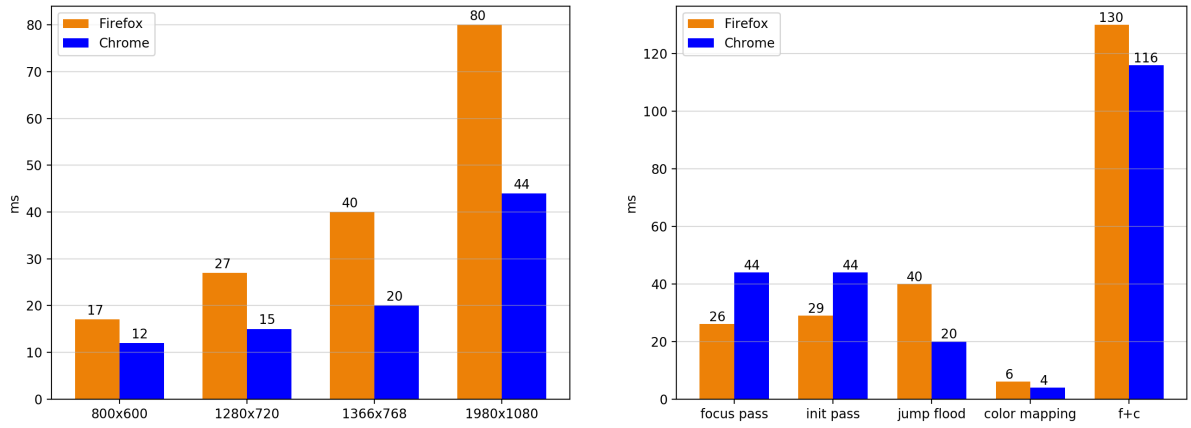
Figure 8: Comparison of the isochrones computed (a) by the presented client-side IBI and (b) by the existing server-side GBI.

4.2 Rendering Performance

Test Data and Hardware. The data to be rendered is a tiled glTF response [23] containing 659 186 vertices and 329 593 line primitives. All tiles are loaded into a tile cache before executing the rendering operations. The WebGL rendering performance measurements are performed on a Microsoft Surface Pro 4 equipped with an Intel Core i7-6650U 2.2 GHz Processor and 16 GB RAM at 1867 MHz. The GPU is an integrated Intel Iris Graphics 540 with 128 MB dedicated VRAM and 8154 MB shared system memory. The respective measurements are performed in full-screen mode on the Surface Display with the half of the native resolution of 2736×1824 pixels at 267 ppi. The test application operates on Microsoft Windows 10 Pro operating system with a 64 bit architecture. The utilized browsers are Mozilla Firefox (Version 55.0.3, Gecko renderer) and Google Chrome (Version 60.0.3112.113, Blink renderer), both with WebGL 1.0/GLSL ES 1.0 support.

Test Procedure. This work focuses on the frontend rendering, which is why the performance evaluation is limited to the measurement of WebGL draw calls for the JFA and the individual rendering passes. Due to the postulated independence of the JFA on the number of input seeds [21], we compare the performance of the jump flooding passes among different screen sizes, as the JFA is stated to be dependent on the texture size. To obtain meaningful results, the JFA draw call is wrapped in a loop, executing 500 times and accordingly being summed and averaged. Since the JFA is a round-based algorithm that updates its input (i.e., textures and step size) in every round, these values have to be reset after every sample measurement to ensure the JFA to operate properly by using the same input every time. Fig. 9a shows the plots of the measurements of the JFA for the two respective browsers.

Test Results. Despite Google Chrome performs constantly better than Mozilla Firefox, the dependency of the speed on the screen size becomes obvious. However, since the isochrone generation is likely to have no need for real-time responses as they are computed only on user-specific map interaction (setting of markers, zooming), this behavior is not of outstanding importance. Instead, the use of caching logic allows for an increasing speed of the JFA, since there is less space for the seed points to flood into. Fig. 9b compares the rendering speed of the prominent rendering passes; the rightmost bars accumulate the others. A behavior worth mentioning is the inferior performance of Google Chrome compared to Firefox in the focus and initialization passes. This seems to be provoked by CPU-intense operations such as matrix transformations or leaflet-related procedures. In contrast thereto, in the JFA and color mapping passes, Chrome outperforms Firefox, since there are no matrix computations to be performed.



(a) Jump flooding rendering performance.

(b) Comparison of rendering passes.

Figure 9: Measurement results of rendering performances with the browsers Mozilla Firefox and Google Chrome. Measurements are performed for (a) the jump flooding algorithm and (b) the rendering passes executed to create Focus and Context views. The performance is measured in milliseconds (ms).

4.3 Limitations and Future Research

Conceptual and Technical Limitations. The proposed technique aims to enable users to effectively extract relevant information from a reachability map. However, due to a considerable amount of pixels representing street segments in the focus area at low zoom levels, one can hardly distinguish individual streets, particularly where the network density is high. The strengths of the NBV [23] thus become apparent primarily at high zoom levels, whereas IBV are sufficient for low zoom level applications. Remedy may be providing a combination with a magnification lens or rather a fish-eye-like projection [20].

The interactive mouse lens (Fig. 6a) allows for a user-defined determination of a ROI, while the isochronal lens (Fig. 6c) attempts to anticipate the relevance of the data by assuming it to be a function of reachability. However, the general purpose of a reachability map is to support the user in yet determining a ROI. Since the approach cannot predict the user's specific interest, the isochronal lens as well as the marker lens (Fig. 6b) are biased in some respects, as a region or object of the user's interest might possibly be located in areas difficult to reach.

Another aspect to consider when dealing with isochrones is the cartographic generalization implicitly applied due to the definition of Voronoi diagrams. As a result, travel time values may be assigned to parts of the map where inherently no information is available and may therefore cause interpretation errors. Technical limitations are furthermore introduced according to the nature of the JFA and the tiling principle, which causes streaks during the asynchronous image composition stage. Besides, the WebGL technology and its extensions lack cross-browser availability.

Future Research Directions. To prove the acceptability of the proposed approach, further user studies have to be conducted. Though the presented work focuses on reachability analytics and a respective visualization using reachability maps, the proposed technique of creating IBI using JFA is not limited to this subject, as it is able to propagate arbitrary information derived from different geographic feature representations beyond networks and can be used in a wider range of applications. Furthermore, another post-processing steps could be envisioned to obtain smoother shapes for the isochrones.

5 CONCLUSIONS

This paper introduces a novel approach to apply lens-based FCV techniques to interactive web-based reachability maps. It presents three types of lenses to provide the user with both, a detailed and a contextual representation of reachability information within a single view and supports the user to extract detailed reachability information while retaining contextual overview. To attain an interactive and responsive application, the web-based implementation takes advantage of graphics hardware using the WebGL standard for GPU programming. Comparisons with geometry-based isochrone generation methods indicate the exchangeable applicability of the presented image-based isochrones.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the paper. This work was funded by the Federal Ministry of Education and Research (BMBF), Germany within the "MOBIE" project (www.mobie-project.de).

REFERENCES

[1] Maneesh Agrawala and Chris Stolte. Rendering effective route maps: Improving usability through generalization. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 241–249, New York, NY, USA, 2001. ACM.

[2] Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michał Iwanicki, and Sébastien Hillaire. *Real-Time Rendering 4th Edition*. A K Peters/CRC Press, Boca Raton, FL, USA, 2018.

[3] Harvey W. Armstrong. A network analysis of airport accessibility in south hampshire. *Journal of Transport Economics and Policy*, pages 294–307, 1972.

[4] Staffan Björk, Lars Erik Holmquist, and Johan Redström. A framework for focus+context visualization. In *Proceedings of the 1999 IEEE Symposium on Information Visualization*, INFOVIS '99, pages 53–, Washington, DC, USA, 1999. IEEE Computer Society.

[5] Tom Carden. Travel time tube map, 2006. Accessed: 2019-04-40.

[6] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1):2:1–2:31, January 2009.

[7] Georg Fuchs, Matthias Kreuseler, and Heidrun Schumann. Extended focus & context for visualizing abstract data on maps. In *CODATA Prague Workshop - Information Visualization, Presentation, and Design*, 03 2004.

[8] G. W. Furnas. Generalized fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '86, pages 16–23, New York, NY, USA, 1986. ACM.

[9] Francis Galton. On the construction of isochronic passage-charts. In *Proceedings of the Royal Geographical Society and Monthly Record of Geography*, volume 3, pages 657–658. JSTOR, 1881.

[10] Jan-Henrik Haunert and Leon Sering. Drawing road networks with focus regions. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2555–2562, dec 2011.

[11] Helwig Hauser. *Scientific Visualization: The Visual Extraction of Knowledge from Data*, chapter Generalizing Focus+Context Visualization, pages 305–327. Springer Berlin Heidelberg, 2006.

[12] Henning Hollburg, Christoph Sinn, and Patrick Voland. *Angewandte Geoinformatik 2012.*, chapter Hier bin ich - was kann ich erreichen? Webbasierte, interaktive Erreichbarkeitsanalyse touristischer Ziele der Stadt Potsdam, pages 317–322. Herbert Wichmann Verlag, VDE VERLAG GMBH, Berlin/Offenbach, 2012.

[13] Ishantha Lokuge; Suguru Ishizaki. Geospace: An interactive visualization system for exploring complex information spaces. *Proceedings of the ACM CHI '95 Conference on Human Factors in Computing Systems*, pages 409–414, 1995.

[14] P. Karnick, D. Cline, S. Jeschke, A. Razdan, and P. Wonka. Route visualization using detail lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):235–247, March 2010.

[15] Johannes Kopf, Maneesh Agrawala, David Bargeron, David Salesin, and Michael Cohen. Automatic generation of destination maps. *ACM Trans. Graph.*, 29(6):158:1–158:12, December 2010.

[16] Robert Kosara, Helwig Hauser, and Donna L. Gresh. An Interaction View on Information Visualization. In *Eurographics 2003 - STARs*. Eurographics Association, 2003.

[17] Josua Krause, Marc Spicker, Leonard Wörteler, Matthias Schäfer, Leishi Zhang, and Hendrik Strobelt. Interactive visualization for real-time public transport journey planning. In *Proceedings of SIGRAD 2012*, number 81, pages 95–98. Linköping University Electronic Press; Linköpings universitet, 2012.

[18] Meinhard Lemke, Carsten Schürmann, Klaus Spiekermann, and Michael Wegener. *Nationalatlas Bundesrepublik Deutschland - Verkehr und Kommunikation*, chapter Transeuropäische Verkehrsnetze, pages 42–45. Institut f. Länderkunde, 2001.

[19] Vincent Meertens. Featured graphic. travel time maps of urban areas in the netherlands. In *Environment and Planning A - Volume 44*, 2012.

[20] Emmanuel Pietriga, Olivier Bau, and Caroline Appert. Representation-independent in-place magnification with sigma lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):455–467, May 2010.

[21] Guodong Rong and Tiow-Seng Tan. Jump flooding in gpu with applications to voronoi diagrams and distance transform. In *I3D '06 Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, 03 2006.

[22] Takafumi Saito and Tokiichiro Takahashi. Comprehensible rendering of 3-d shapes. *SIGGRAPH Comput. Graph.*, 24(4):197–206, September 1990.

[23] Alexander Schoedon, Matthias Trapp, Henning Hollburg, and Jürgen Döllner. Interactive Web-based Visualization for Accessibility Mapping of Transportation Networks. In *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers*, EuroVis '16, pages 79–83, Goslar Germany, Germany, 2016. Eurographics Association.

[24] Amir Semmo, Matthias Trapp, Jan Eric Kyprianidis, and Jürgen Döllner. Interactive visualization of generalized virtual 3d city models using level-of-abstraction transitions. *Comput. Graph. Forum*, 31(3pt1):885–894, June 2012.

[25] Klaus Spiekermann. Visualisierung von eisenbahnreisezeiten - ein interaktives computerprogramm. Final project report, Institut für Raumplanung, Universität Dortmund, 1999. german.

[26] Conrad Thiede, Georg Fuchs, and Heidrun Schumann. *Smart Lenses*, pages 178–189. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[27] Christian Tominski, Stefan Gladisch, Ulrike Kister, Raimund Dachselt, and Heidrun Schumann. A Survey on Interactive Lenses in Visualization. In R. Borgo, R. Maciejewski, and I. Viola, editors, *EuroVis - STARs*. The Eurographics Association, 2014.

[28] Matthias Trapp. *Interactive Rendering Techniques for Focus+Context Visualization of 3D Geovirtual Environments*. PhD thesis, Hasso-Plattner-Institut, Mathematisch-Naturwissenschaftliche Fakultät, Universität Potsdam, 2013.

[29] Matthias Trapp, Christian Beesk, Sebastian Pasewaldt, and Jürgen Döllner. Interactive Rendering Techniques for Highlighting in 3D Geovirtual Environments. In *Advances in 3D Geo-Information Sciences*, volume 9 of *Lecture Notes in Geoinformation and Cartography*, pages 197–210. Springer Berlin Heidelberg, January 2011.

[30] Matthias Trapp, Amir Semmo, and Jürgen Döllner. Interactive rendering and stylization of transportation networks using distance fields. In *Proceedings of the 10th International Conference on Computer Graphics Theory and Applications*, GRAPP 2015, pages 207–219, Portugal, 2015. SCITEPRESS.

[31] Yu-Shen Wang and Ming-Te Chi. Focus+context metro maps. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2528–2535, Dec 2011.

[32] Shi Yin, Moyin Li, Nebiyou Tilahun, Angus Forbes, and Andrew Johnson. Understanding transportation accessibility of metropolitan chicago through interactive visualization. In *Proceedings of the 1st International ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics*, UrbanGIS'15, pages 77–84, New York, NY, USA, 2015. ACM.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Visualizing Time Series Consistency for Feature Selection

Lena Cibulski Thorsten May
Fraunhofer IGD
Fraunhoferstr. 5
64283 Darmstadt, Germany
{first}.{last}@igd.fraunhofer.de

Bernhard Preim
OvGU Magdeburg
Universitätsplatz 2
39106 Magdeburg, Germany
bernhard.preim@ovgu.de

Jürgen Bernard Jörn Kohlhammer
TU Darmstadt
Fraunhoferstr. 5
64283 Darmstadt, Germany
{first}.{last}@gris.tu-darmstadt.de

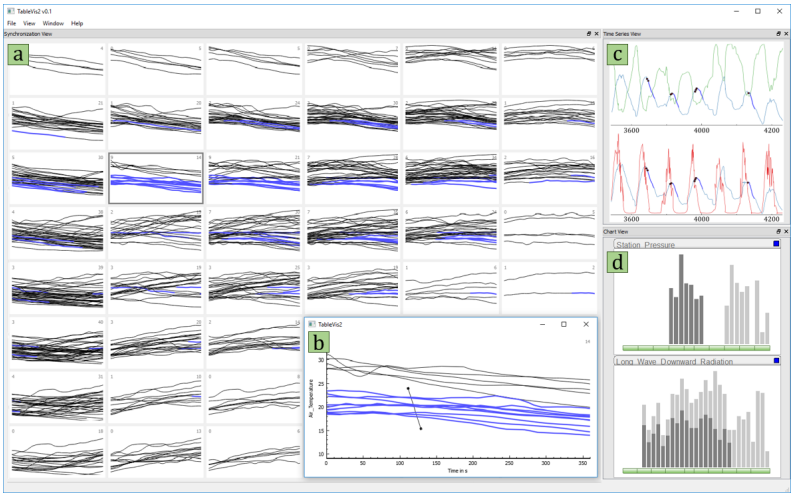


Figure 1: Visualizing how consistently a variable subset captures the characteristics of a dependent variable. Segments corresponding to groups of comparable values on the subset are investigated (a). An inconsistency is brushed (b, c). Histograms (d) show that one of the variables (top) might add to the variable subset's discriminating ability.

ABSTRACT

Feature selection is an effective technique to reduce dimensionality, for example when the condition of a system is to be understood from multivariate observations. The selection of variables often involves a priori assumptions about underlying phenomena. To avoid the associated uncertainty, we aim at a selection criterion that only considers the observations. For nominal data, consistency criteria meet this requirement: a variable subset is consistent, if no observations with equal values on the subset have different output values. Such a model-agnostic criterion is also desirable for forecasting. However, consistency has not yet been applied to multivariate time series. In this work, we propose a visual consistency-based technique for analyzing a time series subset's discriminating ability w.r.t. characteristics of an output variable. An overview visualization conveys the consistency of output progressions associated with comparable observations. Interaction concepts and detail visualizations provide a steering mechanism towards inconsistencies. We demonstrate the technique's applicability based on two real-world scenarios. The results indicate that the technique is open to any forecasting task that involves multivariate time series, because analysts could assess the combined discriminating ability without any knowledge about underlying phenomena.

Keywords

Visual Analytics, Feature Selection, Consistency, Multivariate Time Series, Model-Agnostic, Forecasting

1 INTRODUCTION

Predictive modeling is the process of analyzing historical observations to make a statement about the future. It is called *forecasting* when the time dimension is used as an explicit source of information. As an example, the missing values of a broken sensor can be expressed by a subset of remaining sensors in the network. With multiple sensors available, choosing a representative subset

of input variables becomes a primary challenge. In machine learning, this is referred to as *feature selection*.

Feature selection techniques can be grouped according to the selection criterion used to rate a variable subset. Wrapper methods use a predictive model that is built from the subset. When the type of model can be determined, they usually yield well-performing variable subsets. However, solid indications that guide the decision for a model type might not be available. In such cases, filter methods, which explore data-based characteristics to evaluate a variable subset, might be more feasible.

Among the available filter criteria, most of which rely on combined bivariate or trivariate dependencies, *consistency* is the only one to combine being multivariate by design with two important benefits: *1)* it is model-agnostic, i.e. does not involve any assumption about the nature of data characteristics, and *2)* it removes both redundant and irrelevant variables, thus serving simplicity. Consistency has originally been defined for classification with nominal data. The idea is to identify a minimal variable subset that is consistent with a set of observations. A variable subset is inconsistent, if one or more pairs of observations exhibit equal values for the variables in the subset, but have different class labels. The more inconsistencies a variable subset induces, the worse its relevance for prediction.

However, consistency has not yet been considered in the context of forecasting. Most selection criteria for time series contain an implicit model specification, e.g. using principal components or pairwise cross correlation. This motivates us to investigate how far a *model-agnostic* evaluation of variable subsets for forecasting can get. We do not aim at a new modeling approach. Instead, we focus on feature selection as a preparatory step that guides the following modeling stages.

In this work, we propose a visual-interactive selection technique centered around a criterion that builds upon the consistency of temporal output progressions. To adapt the concept of consistency to time series, we establish counterparts for *1)* the similarity of observations and *2)* the similarity of corresponding outputs. Conceptually, we define an inconsistency as a pair of time points exhibiting similar value combinations with respect to the variable subset, but being followed by dissimilar temporal progressions of the output variable.

Our focus is on leveraging the capabilities of well-established visual analytics techniques to determine how consistently a variable subset captures multivariate time series data. For judging the similarity on the output side, we prefer visual perception over a quantified similarity measure to meet the data- and application-dependent understanding of similarity. The output progressions associated with comparable time points are plotted as aligned curves in a line chart. Perceived inconsistencies indicate portions of the output variable that the investigated variable subset cannot explain. We combine different compositions of line charts with interaction techniques to steer the analysis and to support a refinement of variable subsets (see Figure 1). The contributions of this paper include:

- A consistency-based, model-agnostic selection criterion for multivariate time series
- Overview visualizations conveying the consistency of output progressions across the subset domain
- Detail visualizations and interaction concepts that support the analysis and resolving of inconsistencies

2 RELATED WORK

Our visual-interactive approach intersects the areas of feature selection, regression and predictive modeling, and the analysis of time series. Our literature review shows a lack of feature selection techniques that are both model-agnostic and targeted at temporal data (Table 1). Instead, temporal dependencies are often embedded in similarity or correlation metrics.

2.1 Feature Selection

Molina et al. provide an overview of selection criteria and search strategies for feature selection [MBN02]. Blum and Langley distinguish feature selection methods based on the selection criterion [BL97]. In their terms, our work is most closely related to filters. Filter criteria are usually based on measures for distance, information, dependence, or consistency. Unlike the first three, where scalability is mostly achieved by combining bivariate dependencies, consistency can evaluate a subset of variables at a time, independent of its size. Dash and Liu find consistency to be fast and able to remove redundant variables [DL03]. Sips et al. were among the first to adopt consistency for 2D-projections of labeled numerical data [SNLH09]. However, the concept of consistency has not yet been adapted to time series, and we are aiming to close this gap.

Most visual approaches to feature selection are based on correlation measures. Guo details pairwise correlations in a matrix view to explore interesting variable subspaces for clustering [Guo03]. Krause et al.'s *SeekAView* enables a dual exploration of subspaces and item subsets [KDFB16]. *Infuse* by Krause et al. evaluates the performance of combinations of feature selection and classification techniques [KPB14], essentially integrating filter and wrapper approaches. The *Smartstripes* approach by May et al. turns an automated feature selection into a white-box approach, allowing the user to intervene in an automated heuristic [MBD+11].

Feature selection techniques have also been applied to multivariate time series. Yoon et al. propose *CLeVer*, a ranking feature selection based on a principal component analysis [YYS05]. It effectively eliminates vari-

Method	Model-Agnostic	Time dependence	Partitioning	Expertise	Ranking
Our approach	✓	✓	✓	✓	
CLeVer [YYS05]		✓			✓
Incremental cross correlation [Bac16]		✓			✓
SmartStripes [MBD+11]	✓		✓	✓	✓
Partition-Based Framework [MP13]			✓	✓	✓
Interactive Feature Selection [Guo03]	✓			✓	
Infuse [KPB14]				✓	✓

Table 1: An overview of reviewed methods. Only our approach is model-agnostic *and* supports time dependence.

able redundancies, but does not consider time dependence. In contrast, Bacciu proposes a cross correlation approach that respects the temporal order [Bac16].

2.2 Regression Modeling and Forecasting

Predicting future values based on multivariate time series relates to aspects of both forecasting and regression. Regression models relationships among numerical variables. We contrast forecasting approaches that model relationships between different points in time. These two tasks actually complement each other and can often be performed with the same modeling techniques. Mühlbacher and Piringer present a comprehensive regression approach integrating variable ranking, localized and incremental adaption, and residual analysis [MP13]. In a recent approach, Matkovic et al. inject temporal dependencies into the regression model by means of scalar aggregates [MAJH17]. *TimeFork* by Badam et al. combine regression and forecasting represented by two neural networks for temporal and conditional predictions [BZS+16]. The link between the two models is continuously refined via user interaction and feedback. In principle, all of these approaches allow for choosing a model type, but the decision still has to be made prior to analysis and cannot be postponed.

2.3 Time Series Analysis

Our work is also related to visual analysis of time series ensembles. Konyha et al. present an ensemble visualization to analyze the influence of control parameters on the behavior of dependent time series [KMG+06]. In contrast, we observe how dynamic properties of input time series influence the dependent time series' behavior. Buono et al. propose an approach for the visual analysis of sample-based forecasts [BPS+07]. Like them, we filter variables to produce a comparable subset of samples, but we aim to systematically cover *all* potential filter settings. Schreck et al. use Self Organizing Maps to cluster time series, allowing for a scalable exploration of behaviors [SBvLK09]. However, the approach relies on the definition of a similarity metric, which requires assumptions about the underlying relations. The *TimeCurves* approach by Bach et al. reveals patterns of temporal evolution by distorting the temporal trajectory such that spatial proximity indicates similarity of events [BSH+16]. Instead of keeping the time series connected, we isolate chunks that belong to the same region of the variable subset domain.

Multivariate time series can be aligned to match with discrete events, providing a better frame of reference for comparisons. In *Lifeflow* [WGGP+11], discrete event series are aligned, while *CareCruiser* [GAK+11] aligns continuous time series to discrete events. Our approach also allows alignment to non-discrete events.

3 CONSISTENCY-BASED CRITERION

We first illustrate the concept of consistency-based feature selection on the basis of nominal data. We then explain the challenges of applying this concept to time series and describe our adapted consistency criterion.

3.1 Consistency for Nominal Data

A model approximates a function that maps some input variables $X = \{X_1, ...X_n\}$ to an output variable Y. The exact mapping is unknown, except for a collection of observations $(\vec{x}, y)$ with inputs $\vec{x} = (x_1, ..., x_n); x_i \in X_i$ and outputs $y \in Y$. Any attempt to approximate the mapping needs to capture the input-output assignments as specified in these observations. This requires that no input is associated with several different outputs. We thus search for a subset $X' \subset X$ of variables that fully distinguish among the outputs in the observations. We favor subsets defined over as few variables as possible, also referred to as *Min-Features bias* [AD91].

A variable subset X' allows for distinction of outputs, if no two observations with the same values on the subset have different outputs:

$$\forall \vec{x_1}', \vec{x_2}' \in X' : \vec{x_1}' = \vec{x_2}' \implies y_1 = y_2 \qquad (1)$$

In contrast, outputs are not distinguishable by the variables in X', if two observations exhibit the same values on the subset, but are associated with different outputs:

$$\exists \vec{x_1}', \vec{x_2}' \in X' : \underbrace{\vec{x_1}' = \vec{x_2}'}_{\text{input comparison}} \wedge \underbrace{y_1 \neq y_2}_{\text{output comparison}} \qquad (2)$$

In this case, the output for input $\vec{x_1}' = \vec{x_2}'$ is ambiguous, thus introducing *inconsistency*. In the context of classification, an inconsistency is defined as "two instances [...] that are equal when considering only the [variables] in X' and that belong to different classes" [MBN02]. For deterministic models assigning exactly one output to each input, the ambiguity associated with inconsistencies always results in a modeling error. Consistency criteria therefore aim at inducing as few inconsistencies as possible using the smallest possible subset.

3.2 Adapting Consistency to Time Series

In this section, we describe our approach for transforming the concept of consistency into a selection criterion for multivariate time series data. Each observation $(\vec{x}, y)$ is now associated with a time stamp t. If the temporal order of these observations was ignored, they could be treated just like tabular data. However, this is not an option for an analysis of real-world scenarios.

Instead, we focus on dependencies between $\vec{x}$ and y that manifest across multiple time steps. In theory, the output value to be predicted at a certain time point might be affected by any input variable at any earlier time, although the effect of values might be larger for recent time points. Still, it is not feasible to search this space due to the huge number of variables and time points to be considered. Following the idea of consistency as described for nominal data (Section 3.1), such a dependency can be approached from a different perspective: inputs are considered as the starting point, whose effect on the output values at later times is investigated.

Just like before, we consider observations with the same values on the variable subset. Instead of classes as with nominal data, we now observe the dependent variable's temporal progressions $y : [t, t+l]$ over intervals $[t, t+l]$ on the output side. Thus, we re-define inconsistency as:

$$\exists t_a, t_b : \underbrace{\vec{x}'(t_a) \sim \vec{x}'(t_b)}_{\text{input comparison}} \wedge \underbrace{y : [t_a, t_a+l] \not\sim y : [t_b, t_b+l]}_{\text{output comparison}} \tag{3}$$

Contrary to Equation (2), we consider input similarity rather than equality, because we cannot expect two observations to exhibit the exact same numbers for the numerical variables in the subset. Given Equation (3), a variable subset is said to be consistent if, by means of the inputs $\vec{x}'$ specified by the involved variables, dissimilar output progressions can be distinguished.

3.3 Analysis Tasks

To evaluate a variable subset using the adapted consistency criterion, Equation (3) is applied to multivariate time series. This is guided by the following questions:

- *Input comparison* – are numerical observations similar regarding the variable subset?
- *Output comparison* – are output progressions associated with similar inputs similar as well?
- *Inconsistency search* – where do outputs progress inconsistently across the variable subset domain?
- *Inconsistency analysis* – under which conditions do they occur? How can they be resolved?

Based on these leading questions, we propose the following tasks to be addressed by our visual technique to perform one iteration of the feature selection procedure:

- $\mathbf{T}_1$ Grouping of input vectors
- $\mathbf{T}_2$ Visualizing outputs for individual input groups
- $\mathbf{T}_3$ Layouting of visualizations for all input groups
- $\mathbf{T}_4$ In-depth analysis of inconsistencies

4 VISUAL FEATURE SELECTION

In the following, we present the role of interactive visualization to address the tasks of consistency-based feature selection as explained in Section 3. They apply to the subset evaluation and candidate generation in each iteration of the procedure. Note that the increasing dimensionality of the variable subset affects the part of an observation that we consider as input.

Both input and output comparisons require an understanding of similarity. For the input space, we present a grouping strategy that subdivides the value domain of a variable subset into disjoint regions of similar inputs (Section 4.1). The actual challenge lies in representing the task-dependent understanding of output similarity. The output progressions for each region are aligned for visual comparison in a dedicated chart (Section 4.2). The inconsistency across regions is conveyed by small multiples arranged in 2D visual space (Section 4.3). Finally, we provide interaction concepts for in-depth analysis and elimination of inconsistencies (Section 4.4).

4.1 Grouping

As described in Section 3.2, a variable subset is evaluated in terms of its consistency. It involves the accumulation of inconsistencies for all possible expressions of inputs as occurring in the data. Before we can assess inconsistencies, we thus need to perform a *grouping* of inputs to unique expressions ($\mathbf{T}_1$).

A straight-forward solution to grouping is an equidistant binning strategy as it is known from histogram generation. For two or more variables, we use a regular grid, which corresponds to the Cartesian product of the individual binnings. The binning resolution can be adjusted at any point of the analysis. A finer resolution represents the input domain more accurately, but might lead to more regions being empty or containing a statistically insignificant number of time points. Regardless of the number of variables, the result of grouping are sets of time points where inputs are highly similar.

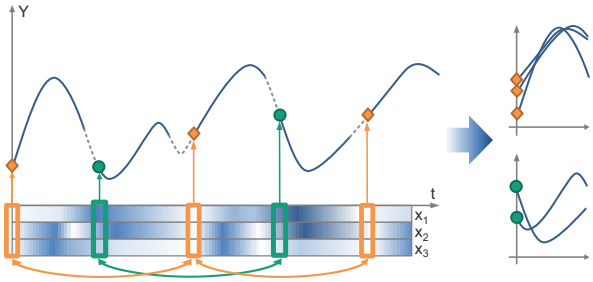


Figure 2: Visual comparison of output progressions. Synchronization points mark comparable inputs (left). The output progressions are cropped and shifted along the time axis to establish a common baseline (right).

We choose binning as a simple, yet effective solution, because it does not involve assumptions. Additional benefits include: *1)* axis-orthogonal partition boundaries are more easily translated into a visual representation than free-form regions, *2)* axis-orthogonal boundaries allow for a description of individual regions by means of value ranges, and *3)* it is free of unnecessary functionality such as maximizing inter-group distances.

4.2 Visualization of Individual Groups

The binning yields a discretization of the variable subset domain according to the similarity of observations. The purpose of visualization is to expose patterns of those output progressions that correspond to each group of similar inputs ($\mathbf{T}_2$). To make the progressions comparable, we propose the following alignment technique.

Synchronization The synchronization takes inputs that share common characteristics with respect to the variable subset and aligns the corresponding output progressions. It is performed in three steps (Figure 2):

1) Find all time points at which the given inputs arise
2) Split the output time series at these points and crop the sections to an interval length l
3) Shift the cropped sections on top of each other

Comparable inputs arise at different time points t_i, which we call *synchronization points* (Figure 2, left). As noted in Equation (3), the outputs to be compared are the progressions $y : [t_i, t_i + l]$ that evolve from the synchronization points onward. Under the premise that all t_i represent a common state, consistent progressions should be similar in relation to the synchronization points. We establish a common baseline by shifting the t_i as well as the progressions bound to them along the time axis, such that the synchronization points are mapped to $t = 0$. This is what we call *synchronization*. The result is depicted in Figure 2, right.

Synchronization Chart The *Synchronization Chart* depicts the output progressions obtained from the synchronization as an ensemble of line segments ($\mathbf{T}_2$). In contrast to traditional line charts, the time axis does not represent absolute times but relative to the synchronization points. The chart allows analysts to gain insight into inconsistencies among the output progressions. What is considered similar may vary depending on the nature of the data, the application, and the analysis focus. Providing an adequate visual representation allows analysts to take advantage of their visual understanding of similarity.

Figure 3 shows a collection of patterns indicating different forms of the relation between a variable subset and the output variable. The most meaningful patterns to be observed are highly similar output progressions (Figure 3a) or arbitrary behavior (Figure 3b). These two cover the entire spectrum from consistency to inconsistency. Another example are two diverging sets of progressions (Figure 3c). The common starting point indicates a relation between variable subset and output at least at the synchronization point. However, the subset seems to be missing a variable that discriminates between positive (green) and negative (purple) slope of the progressions. Another example are contrary behaviors (Figure 3d). The symmetry of the two groups and the uniformity of progressions within a group indicate that the variable subset already captures the output behavior quite well. The remaining separation of the groups might be done based on the output values at the synchronization point.

4.3 Layouting

For each non-empty group of inputs resulting from binning, the synchronized output progressions are visualized in a chart. What is missing for a broad assessment of consistency across the variable subset domain is a layouting strategy that arranges all charts in an overview visualization ($\mathbf{T}_3$). This strategy needs to cope with a common problem in the visualization of multivariate data: an arrangement in 2D visual space cannot preserve the positions of the groups in nD space.

To perceive how consistently a variable subset captures the output characteristics, it is more important to observe the spread of progressions within each group of inputs than between groups. This allows for a simplification of the visualization at two levels: *1)* the presentation of charts and *2)* their arrangement. For the presentation, we omit axes and labels of the charts to reduce visual clutter and to address the limited screen space available for each chart. Regarding the arrangement of the resulting small multiples, small errors in the representation of the topology in n-dimensional subset space are acceptable, as the focus lies on within-group comparisons of progressions. Thus, we trade a non-exact topology preservation for an efficient arrangement of charts, meaning simple and not costly to compute.

Finally, the layouting strategy is determined by the number of variables in the subset. For one or two variables, the input space can be directly mapped to

(a) Similar behaviors (b) Arbitrary behaviors (c) Diverging behaviors (d) Contrary behaviors

Figure 3: Patterns of output progressions convey different notions of consistency: similar progressions indicate consistency (a), arbitrary output behaviors indicate inconsistency (b), a shared origin indicates consistency at least at the synchronization point (c), distinguishable behaviors indicate the need for a discriminating variable (c,d).

Figure 4: Synchronization Charts depict the output progressions across the value domain of a single variable. Despite the divergence, similarity indicates consistency.

the visual space. For one variable, charts are concatenated to a *Synchronization Row* (Figure 4), representing the intervals that result from binning. For two variables, the charts are arranged in a *Synchronization Grid* (Figure 5), which corresponds to the regular grid used for binning. Higher-dimensional subsets require additional considerations concerning the reduction to the two-dimensional visual space. The most pragmatic layouting strategy is to use dimension reduction. Even if the topology of inputs cannot be completely preserved, a layout based on dimension reduction might help to steer the analysis in a meaningful way.

4.4 In-Depth Analysis of Inconsistencies

The overview visualization enables analysts to perceive regions, where output progressions are dissimilar. However, the mere existence of inconsistencies does not always indicate an incomplete variable subset. An inconsistency may occur for two reasons: either *1)* the variable subset is missing some relevant variables or *2)* the underlying inputs, too, vary over time. Inconsistencies caused by the latter are considered false positives. The next step towards a refinement of the variable subset is to examine the circumstances under which the inconsistency occurs to eliminate false positives ($\mathbf{T}_4$).

Focus Visualization and Brushing To avoid similarity metrics wherever possible, our technique heavily relies on an interactive definition of what is perceived as (dis-)similar. Having identified an inconsistency in the overview visualization, analysts can bring the corresponding small multiple to focus. It then turns into an enlarged Synchronization Chart with axes and labels for orientation (Figure 5, center). This *Focus Plot* allows for interaction like zooming, panning, and brushing to investigate output progressions in more detail, while keeping the variable subset domain as a context.

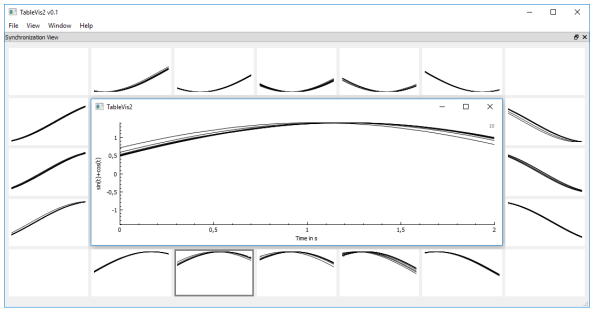


Figure 5: 2D variable subsets are visualized by small multiples in a regular grid. An enlarged chart allows for in-depth analysis of progressions in a region of interest.

From the detailed exploration, the analyst might have identified a number of progressions that make up an inconsistency. To investigate related data characteristics, the corresponding curves are selected. We provide a line brush (Figure 6a) as well as a rectangular brush. During analysis, users can flexibly adjust the mode. Brushing output progressions defines sequences of absolute time points, on which any further analysis is focused. In the following, we describe interaction concepts (Figure 6) that allow analysts to explore different aspects of any perceivable pattern of progressions.

Temporal Context In a Synchronization chart, all output progressions are depicted relative to the synchronization points. An overall temporal context is not given. For analysts, it requires significant cognitive effort to build up a mental image of how the depicted progressions are actually part of the same output time series. Viewing synchronization points and output progressions in a temporal context raises awareness for inappropriate default choices or inconclusive patterns in the output progressions that may otherwise be misinterpreted. We support analysts by linking brushed output progressions to a line chart with respect to *1)* the synchronization points and *2)* the actual progressions (Figure 6b). For *1)*, markers at the respective absolute time coordinates in the line chart indicate the synchronization points. For *2)*, the corresponding sequences of absolute time points are highlighted in the line chart.

Analysis of Input Variations Some output inconsistencies can be explained by variations of the underlying

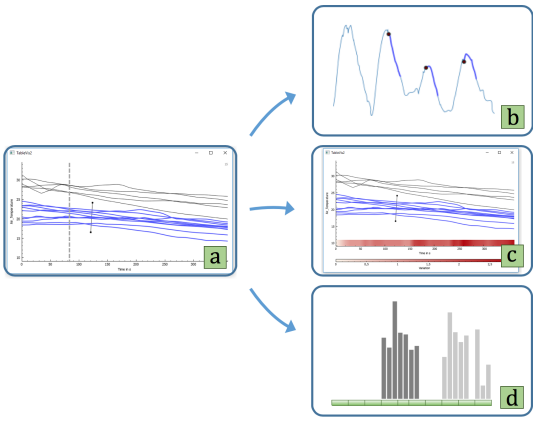


Figure 6: (a) Interaction in the Focus Plot: (b) brushed sections show in the global time range, (c) a one-row heatmap depicts the deviation of inputs, (d) brushed inputs are shown in histograms for candidate generation.

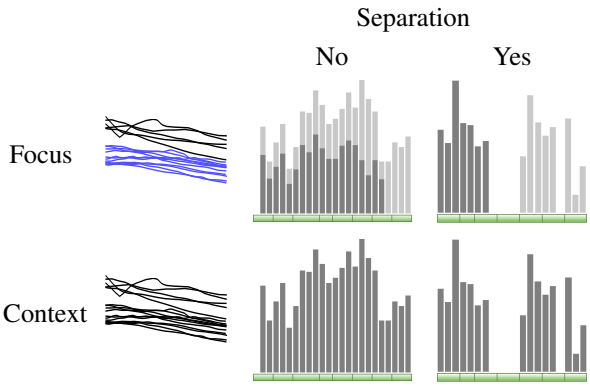


Table 2: Visually assessing the relation of a variable to a brush. Similar distributions indicate that there is none (left). Large differences suggest that the variable separates brushed from non-brushed progressions (right).

inputs. To distinguish these false positives from true inconsistencies, we investigate how the variance of inputs develops along the relative time points in a synchronization interval. Each relative time point is associated with a number of absolute time points. We aim at validating that the input similarity remains unchanged.

This can be performed based on aggregation using the *Variation Indicator*. It is a one-dimensional heat map along the time axis of a Synchronization Chart (Figure 6c). Color depicts the standard deviation of inputs at each relative time coordinate. Little and, if any, slowly increasing deviation indicates that input variation cannot be accounted for variations in the output. Consequently, any conclusion that is drawn from inconsistencies can be focused on variable subset incompleteness. The indicator can be consulted at any stage of the analysis to exclude output inconsistencies that are not necessarily caused by variable subset incompleteness.

Resolving Inconsistencies An inconsistency can be resolved by adding a variable that contributes to a separation between brushed and non-brushed progressions. The best candidates are variables whose values differ exactly where the output progressions differ. For diverging progressions, this could be a variable whose low and high values co-occur with the brushed and non-brushed progressions respectively. Investigating how progressions relate to the values of a variable provides valuable hints on its suitability as additional predictor.

To determine the relation of a candidate variable X_c to a brush, we compare two value distributions:

- *Focus distribution*: those values of X_c that are associated with the brushed observations
- *Context distribution*: those values of X_c that are associated with all observations depicted in the cell

If the shape of the focus distribution follows that of the context distribution, candidate variable and brushed progressions are considered independent (Table 2, left).

Relating inconsistencies to the value characteristics of a candidate variable is supported by depicting the value distributions in a histogram (Figure 6d). If the focus distribution highly differs from the context distribution (Table 2, right), this indicates that the candidate variable is related to the brushed progressions. It means that the variable carries significant potential to separate the brushed progressions from the rest. Including it into the current variable subset might lead to a better discrimination of the output, thus increasing the consistency.

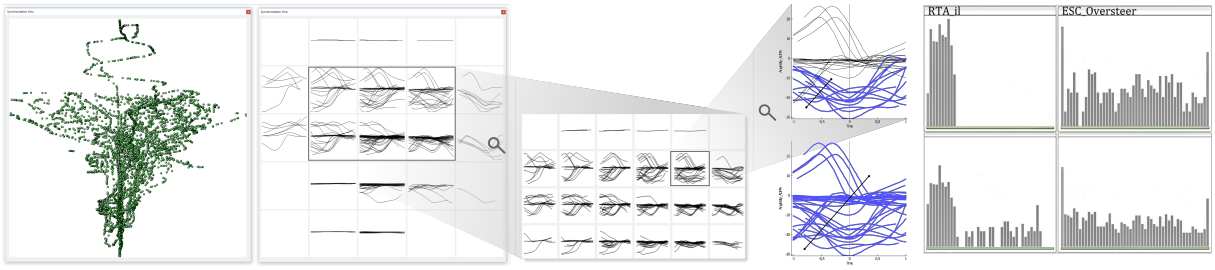
5 PROOF OF CONCEPT

We outline the benefits and limitations of our technique by performing one feature selection iteration using two different multivariate time series data sets. This proof of concept illustrates what can be learned from visualizations and interaction at each stage of the work flow. It aims at demonstrating the potential of visually perceived consistency as a model-agnostic criterion.

5.1 Real-World Meteorology Data

Meteorologists study weather events and climate trends to explain how the atmosphere affects the earth and to predict future weather developments. To investigate the effects of radiation and weather measurements on the temperature, we use data of the Baseline Surface Radiation Network [Beh11]. The data set shown in Figure 1 contains temperature, humidity, air pressure, and radiation measurements from August 2003, when Europe experienced an exceptionally hot summer.

Temperature is defined as the output variable. An initial variable subset is determined from visual exploration of line charts. We observe a negative correlation between *temperature* and *humidity* (Figure 1c, top). Furthermore, we notice that *shortwave-downward radiation* is also related to the *temperature* (Figure 1c, bottom). According to the domain experts, this relation can be explained with cloud-cover vs. clear-sky conditions.



(a) (b) (c) (d) (e)

Figure 7: A feature selection iteration given a 2D variable subset. An overview of the input domain (a) underlying the Synchronization Grid (b) supports the perception of consistency. Increasing the resolution for an inconsistent region hints at the most critical cell (c). The search for additional predictors is driven by distinguishable sets of output progressions (d). Histograms suggest that *RTA_il* might increase consistency when added to the subset (e).

Figure 1a shows the *temperature* progressions evolving from different combinations of *shortwave-downward radiation* and *humidity* within an interval of about six hours. The progressions do not behave arbitrarily, exhibiting an overall consistency across the domain. However, there are also regions of inconsistency exhibiting two distinguishable behaviors (Figure 1b).

Adding a third variable that separates these two types might significantly increase the explanatory power of the current variable subset. Based on the suggestions of the domain experts, we consider *longwave-downward radiation* and *station pressure* as candidates. For in-depth analysis, we brush one of the two separable sets of *temperature* progressions (Figure 1b). For the temporal context, the brushed sections and the synchronization points are highlighted in the line charts (Figure 1c).

Given the brushed set of *temperature* progressions, the histograms for *longwave-downward radiation* and *station pressure* show the corresponding focus distributions in dark-gray, while context distributions are depicted in light-gray (Figure 1d). The differences between focus and context distributions regarding shape and domain coverage in either histogram indicate that both candidate variables are related to the brush. *Station pressure* even achieves a perfect discrimination of the *temperature* behaviors (Figure 1d, top). It is therefore a promising variable to refine the variable subset. According to the experts, *station pressure* reflects the weather conditions connected to the atmosphere, thus relating to *temperature* beyond the daily periodicity of measurements. This confirms our findings.

Note, that these conclusions are drawn from analysis of a small part of the variable subset domain. An outlook of the consistency of the newly generated variable subset could verify the choice before proceeding with the next iteration of the feature selection.

5.2 Real-World Car Sensor Data

We also applied our technique to sensor data from the automotive domain. The data was acquired in the context of vehicle dynamics simulation [Unt13]. Due to a malfunction, the values of the *slip angle* sensor needed to be expressed by a subset of the remaining sensors.

Based on their domain knowledge, the experts suggested two variables to start the analysis with: *yaw rate* and *velocity*. To gain an overview of their value domain, we first take a look at clusters, empty regions, and outliers in a scatter plot (Figure 7a). A first notion of how the output variable behaves across this domain is conveyed in a Synchronization Grid with an initial binning resolution of 5×5 (Figure 7b). Despite a low level of detail, analysts can identify whether output progressions follow certain patterns or behave arbitrarily.

The analysis is driven by regions where output progressions are dissimilar, indicating that – at least for this particular part of the domain – the initial variables do not yet consistently capture all output characteristics. To fix this, we need to investigate the circumstances under which the inconsistency occurs. The drill-down to the most critical cell for detailed analysis requires an increased the grid resolution (Figure 7c).

Adding the variable that most effectively separates the depicted output behaviors might increase the discriminating ability of *{yaw rate,velocity}*. Based on domain knowledge or prior findings, candidate variables have been identified. To verify their suitability as additional predictors, we examine their relation to the brushed output progressions (Figure 7d, top) and the entirety of curves (Figure 7d, bottom) as described in Section 4.4.

For the first variable, the value distribution associated with the brush clearly deviates from the overall distribution (Figure 7e, left), not only in shape but also in domain coverage. Its separating ability makes it a promising variable to be added to the current variable subset. In contrast, the value distribution of the second variable is only marginally influenced by brushing the subset of output progressions (Figure 7e, right). This suggests that it does not provide separating potential and is thus unsuitable as a predictor. Still, such conclusions need to be verified in collaboration with domain experts.

6 DISCUSSION

The most important benefit of our visual feature selection technique is its generality. After selection, the typical decisions to be made for modeling, e.g. model type and parameters, are still available for disposition. In the common case that a model cannot be specified in advance, it provides a meaningful selection of variables rather than searching among different combinations of models and variable subsets in an exhaustive manner.

Unlike most forecasting methods, the proposed criterion does not consider historical patterns on the input side, which might significantly enhance the prediction of output progressions. We plan to address such an extension to the model-agnostic time series criterion in our future research. This might also involve considerations concerning an application to streaming data.

Due to its generality, our technique can be applied to any feature selection task involving multivariate time series. To convey the essential idea of our proposed criterion, we informally evaluated it by means of use cases from the meteorology and automotive domain. The proof of concept showed that, for one iteration, the technique effectively supported the subset evaluation and candidate generation. However, the limitations of our technique, in particular compared to alternative methods, need to be examined by a further evaluation.

While the proof of concept in this paper refers to the meteorology and automotive domain, preliminary tests with artificial data sets also resulted in useful findings. Even though the variable subset had been initialized with distracting variables, the true predictors could be identified in the further course of the feature selection. However, the technique did not produce the minimum subset. The uncertainty about the optimality of the solution is a limitation of virtually all heuristic approaches and our work is no exception.

To fully exploit the multivariate nature of our proposed time series criterion, the input grouping and chart layouting, too, should be multivariate. The Cartesian products of individual variable binnings used for grouping might result in empty or extremely dense regions that impair a statistical analysis of inconsistencies. Additionally, similar inputs might end up on two sides of an interval boundary. Both could be avoided by using adaptive grouping approaches that flexibly determine interval boundaries according to an objective function. The chart layouting is a more challenging task. In principle, any deterministic layout allows for assessing the consistency of a variable subset, even if charts are positioned randomly. However, a topology-preserving layout allows to associate perceived inconsistencies with the depicted portion of the variable subset domain. Approaches like dimension reduction could be combined with force-directed layouts to prevent overlaps. Although our pragmatic strategies could be considered as a limitation for now, both components, the grouping and the layouting, are open to any (not yet realized) scalable technique that suits the data and analysis best.

7 CONCLUSION & FUTURE WORK

In this paper, we presented a visual-interactive consistency-based feature selection technique for multivariate time series data. A key benefit of our technique is the purely model-agnostic evaluation of the combined quality of a variable subset. The criterion is solely based on the input-output characteristics reflected in time-dependent observations. It describes how consistently the subset captures these characteristics. The comparison of temporal progressions necessary for that purpose is solved visually to avoid uncertain assumptions about the nature of the data. Small multiples depict progressions that evolve from comparable inputs across the variable subset domain. An input grouping and layouting strategy present the charts in a way that allows for a detailed analysis of inconsistencies, working toward a refinement of the variable subset under consideration of previous insights. This is supported by interaction concepts that allow analysts to focus on findings, contribute their domain knowledge, and exclude false positive consistencies. Our technique is applicable as a preparatory step to any modeling task involving multivariate time series, in particular when decisions about the model specification shall be postponed.

We identify several directions for future work. The pragmatic approaches to grouping of inputs and layouting of charts could be extended by more scalable techniques to achieve a better representation of higher-dimensional variable subsets. A search strategy dedicated to work with our proposed criterion might provide a valuable contribution to reaching the goal of a minimal variable subset inducing the fewest possible inconsistencies. The strategy might also involve user guidance in the form of a ranking of recommended additional predictors. Finally, we plan to examine the potential of our technique for the validation of existing forecasting models as well as for data quality control. Inconsistencies regarding inputs and model outputs might indicate unsuitable parameter choices or an incomplete variable subset, and thus indicate a questionable model. Inconsistencies might also hint at low data quality, e.g. implausible spikes. In the case of missing data, our technique might help to impute values that are consistent with the observed data.

8 ACKNOWLEDGMENTS

This work was funded by the EC H2020 programme under grant agreement n° 768892.

9 REFERENCES

[AD91] Hussein Almuallim and Thomas G Dietterich. Learning with many irrelevant features. In *Association for the Advancement of Artificial Intelligence (AAAI)*, vol. 91, pp. 547–552, 1991.

[Bac16] Davide Bacciu. Unsupervised feature selection for sensor time-series in pervasive computing applications. *Neural Computing and Applications*, 27(5):1077–1091, 2016.

[Beh11] Klaus Behrens. Basic measurements of radiation at station lindenberg (2003-08), 2011.

[BL97] Avrim L Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997.

[BPS+07] Paolo Buono, Catherine Plaisant, Adalberto Simeone, Aleks Aris, Galit Shmueli, and Wolfgang Jank. Similarity-based forecasting with simultaneous previews: A river plot interface for time series forecasting. In *11th Int. Conf. on Information Visualization*, pp. 191–196, 2007.

[BSH+16] Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, and Pierre Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):559–568, 2016.

[BZS+16] Sriram Karthik Badam, Jieqiong Zhao, Shivalik Sen, Niklas Elmqvist, and David Ebert. Timefork: Interactive prediction of time series. In *Proc. of the CHI Conference on Human Factors in Computing Systems*, pp. 5409–5420, 2016.

[DL03] Manoranjan Dash and Huan Liu. Consistency-based search in feature selection. *Artificial Intelligence*, 151(1-2):155–176, 2003.

[GAK+11] Theresia Gschwandtner, Wolfgang Aigner, Katharina Kaiser, Silvia Miksch, and Andreas Seyfang. Carecruiser: Exploring and visualizing plans, events, and effects interactively. In *IEEE Pacific Visualization Symposium (PacificVis)*, pp. 43–50, 2011.

[Guo03] Diansheng Guo. Coordinating computational and visual approaches for interactive feature selection and multivariate clustering. *Information Visualization*, 2(4):232–246, 2003.

[KDFB16] Josua Krause, Aritra Dasgupta, Jean-Daniel Fekete, and Enrico Bertini. Seekaview: An intelligent dimensionality reduction strategy for navigating high-dimensional data spaces. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, 2016.

[KMG+06] Zoltan Konyha, Kresimir Matkovic, Denis Gracanin, Mario Jelovic, and Helwig Hauser. Interactive visual analysis of families of function graphs. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1373–1385, 2006.

[KPB14] Josua Krause, Adam Perer, and Enrico Bertini. INFUSE: Interactive feature selection for predictive modeling of high dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1614–1623, 2014.

[MAJH17] Krešimir Matković, Hrvoje Abraham, Mario Jelović, and Helwig Hauser. Quantitative externalization of visual data analysis results using local regression models. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pp. 199–218, 2017.

[MBD+11] Thorsten May, Andreas Bannach, James Davey, Tobias Ruppert, and Jörn Kohlhammer. Guiding feature subset selection with an interactive visualization. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, 2011.

[MBN02] Luis Carlos Molina, Lluís Belanche, and Àngela Nebot. Feature selection algorithms: A survey and experimental evaluation. In *Proc. of the IEEE Int. Conf. on Data Mining*, pp. 306–313, 2002.

[MP13] Thomas Mühlbacher and Harald Piringer. A partition-based framework for building and validating regression models. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):1962–1971, 2013.

[SBvLK09] Tobias Schreck, Jürgen Bernard, Tatiana von Landesberger, and Jörn Kohlhammer. Visual cluster analysis of trajectory data with interactive kohonen maps. *Information Visualization*, 8(1):14–29, 2009.

[SNLH09] Mike Sips, Boris Neubert, John P. Lewis, and Pat Hanrahan. Selecting good views of high-dimensional data using class consistency. In *Computer Graphics Forum*, vol. 28, pp. 831–838, 2009.

[Unt13] Michael Unterreiner. *Modellbildung und Simulation von Fahrzeugmodellen unterschiedlicher Komplexität*. PhD thesis, Universität Duisburg-Essen, 2013.

[WGGP+11] Krist Wongsuphasawat, John Alexis Guerra Gómez, Catherine Plaisant, Taowei David Wang, Meirav Taieb-Maimon, and Ben Shneiderman. Lifeflow: Visualizing an overview of event sequences. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1747–1756, 2011.

[YYS05] Hyunjin Yoon, Kiyoung Yang, and Cyrus Shahabi. Feature subset selection and feature ranking for multivariate time series. *IEEE Transactions on Knowledge and Data Engineering*, 17(9):1186–1198, 2005.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Detailed 3D Face Reconstruction from a Single RGB Image

Gemma Rotger[1] grotger@cvc.uab.es
Francesc Moreno-Noguer[2] fmoreno@iri.upc.edu
Felipe Lumbreras[1] felipe@cvc.uab.es
Antonio Agudo[2] aagudo@iri.upc.edu

[1] Computer Vision Center and Departament Ciències de la Computació, UAB
UAB Campus O Building, (08193) Cerdanyola del Vallès, Spain
[2] Institut de Robòtica i Informàtica Industrial, CSIC-UPC
4-6 Llorens i Artigas, (08028) Barcelona, Spain

ABSTRACT

This paper introduces a method to obtain a detailed 3D reconstruction of facial skin from a single RGB image. To this end, we propose the exclusive use of an input image without requiring any information about the observed material nor training data to model the wrinkle properties. They are detected and characterized directly from the image via a simple and effective parametric model, determining several features such as location, orientation, width, and height. With these ingredients, we propose to minimize a photometric error to retrieve the final detailed 3D map, which is initialized by current techniques based on deep learning. In contrast with other approaches, we only require estimating a depth parameter, making our approach fast and intuitive. Extensive experimental evaluation is presented in a wide variety of synthetic and real images, including different skin properties and facial expressions. In all cases, our method outperforms the current approaches regarding 3D reconstruction accuracy, providing striking results for both large and fine wrinkles.

Keywords

3D Wrinkle Reconstruction, Face Analysis, Optimization.

1 INTRODUCTION

Reconstructing the 3D human geometry from RGB images has been extensively studied in computer vision and computer graphics in the past two decades [AMCMN16, AMAC17, GVWT13, GZC+16, LZZL16, SWTC14, ZTG+18, CCC+18]. These results can be applied to many everyday applications from the movie industry to medical purposes, robotics, gaming, or human-computer interaction, to name a few. A significant area of research is focused on the acquisition of human faces. Unfortunately, it is well known that human faces are highly variable, differing widely between gender, age, ethnicity, and gesture expression, making the task harder. For this reason, the study of high-detailed face geometry has transcended amongst other methods.

Face reconstruction from monocular information is known to be a highly under-constrained problem, requiring the use of additional priors to constrain the solution. Early approaches proposed to address the problem through the use of facial 3D morphable methods (3DMM) [BV99, CWZ+14], allowing to recover the large-scale geometry. Thanks to the parametric fitting problem nature, they yield in a considerable dimensionality reduction of the solution space. Nevertheless, due to the generality of these approaches, they are not suitable to retrieve subject-specific details, such as wrinkles and scars, being necessary the use of refining algorithms.

Recently, the use of Deep Learning (DL) strategies has been proposed for face reconstruction from RGB information. In this context, many efforts have been done by proposing 3DMM-based models [RSOEK17], volumetric regression [JBAT17], and learning from synthetic data [RSK16]. While these techniques have proved a good performance, the solutions have a remarkable lack of individual detail even after training specific refining networks. A direct solution would be increasing the amount of training data, exploring the solution space of the wrinkles, which often is not trivial in practice. Notably, most of the before mentioned methods need a refinement step to retrieve individual details.

In this paper, we present a novel optimization framework that combines the use of wrinkle properties, directly extracted from the input image with a photomet-

ric energy term, in order to recover detailed 3D faces from a single image with the highest level of detail possible. On the one hand, we take advantage of current solutions based on deep learning [JBAT17] to obtain a coarse initialization. While this initializing solution cannot recover fine details, it is accurate enough to estimate an initial 3D mesh shape. Recall that we only use deep learning for initialization, i.e., no training data is employed to model wrinkles. On the other hand, we estimate the illumination parameters required later to define a photometric energy term. Finally, we encode wrinkles by using smooth functions. For this purpose, we find wrinkles that can be accurately described in the image up to a depth parameter, since relief caused by wrinkles produce visible shading in the image. We then group the connected pixels after filtering noisy measurements (removing unconnected pixels and pixels with low response values), since they have similar displacement behavior. After that, we operate locally fitting a second-order polynomial that better adapts to the pixel coordinates of each wrinkle detected in the image. With the distance from the pixel to the polynomial curve, the overall wrinkle location, orientation, height, and width, we can model the wrinkle displacement and transfer it to the node which is related to the pixel. We have to take into account that, depending on the mesh and image resolutions, not all the pixels must have a correspondent vertex. As much the pixel-to-vertex correspondences lead to one-to-one, the better the performance.

Our approach offers a realistic solution for high-resolution single image 3D face reconstruction, which differs with current methods since it is both fast and effective, as well as can be easily adapted as a refinement strategy for most of the initializing formulations. We present experimental validation in a wide variety of synthetic and real images, including different skin properties and facial expressions, showing the suitability of our approach even in extreme cases like special effects facial makeup.

2 RELATED WORK

The 3D reconstruction of faces from RGB images has remained an appealing topic. Several methods addressed the problem from different perspectives. First, parametric methods represent a face solution as a linear combination of shape bases, which can be inferred directly from data [SKSS14, AMN19, AMAC17], or predefined in advance [LXC+15]. From the perspective of RGB-D data, other approaches [ZNI+14] required a template to reconstruct a fully rigged face. However, the shape and size variance of the face becomes a limitation to this model. In particular, when the skin produces subject-specific wrinkles and folds on aging and expression, the capture of these medium and fine-scale details compounds the problem.

In the recent years, certain multi-camera [GSSM15], binocular [GFT+11, VWB+12], and monocular approaches [AMN17, AMN18, GVWT13, GZC+16, LZZL16, SWTC14] have obtained remarkable detailed results. Garrido *et al.* [GVWT13] appended detail to a personalized blend shape model, and Shi *et al.* [SWTC14] proposed an iterative process between large-scale reconstruction and fine-scale per-pixel shading cues. More recently, in [GZC+16] has combined shape from shading and learning of a generative detail deformation model, whilst in [LZZL16] proposed to regress a cascaded 3D face shapes model in 2D and 3D spaces. All of them presented accurate solutions over monocular videos but they need a complex setup, as a manual initialization [GVWT13]. Eventually, in [CBZB15] was presented as the first real-time approach by training local regressors to predict wrinkles in monocular sequences. While these approaches produced a great advance in the topic, most of them relied on spatio-temporal priors which become not suitable for single image reconstruction.

Estimating the 3D reconstruction from a single image is a severely under-constrained problem. Early approaches [GMMB00, WWY06, WBS01] were hardly able to recover simple objects like geometric figures and wall-like scenes. The 3D reconstruction of human faces arose with the introduction of 3DMM [BV99, CWZ+14], where the problem was simplified to a low-rank fitting problem, thus not the full geometry was estimated but a set of parameters controlling identity and expression. We can find two different groups in the literature: model-based [HHT+16, JZD+18, RV05] and data-based [RSK16, RSOEK17] approaches. Regarding model-based approaches, they frequently use a non-linear least squares multi-feature fitting, which retrieves a 3D facial structure and the corresponding texture map [RV05], performs under different resolution levels [HHT+16], and fits a photometric minimization energy term to find the parameters that better adjusts the input image [JZD+18]. Even these methods achieve a considerable level of detail, they require plenty of time to offer detailed results. Furthermore, the 3DMM may not yield in a correct result if the given face it is not well represented in the low-rank model.

As in most of the fields, DL approaches attempt to provide an accurate solution to detailed face reconstruction from a single image [JBAT17, RSK16, RSOEK17, TTHM+18, CCL18]. However, we find in DL solutions the lack of a detailed reconstruction –the most obvious of the alleged detailed methods–, since the current DL approaches are not able to directly predict a face with detailed facial features such as wrinkles or scars. In the specific case of [RSK16] they refine their solution by

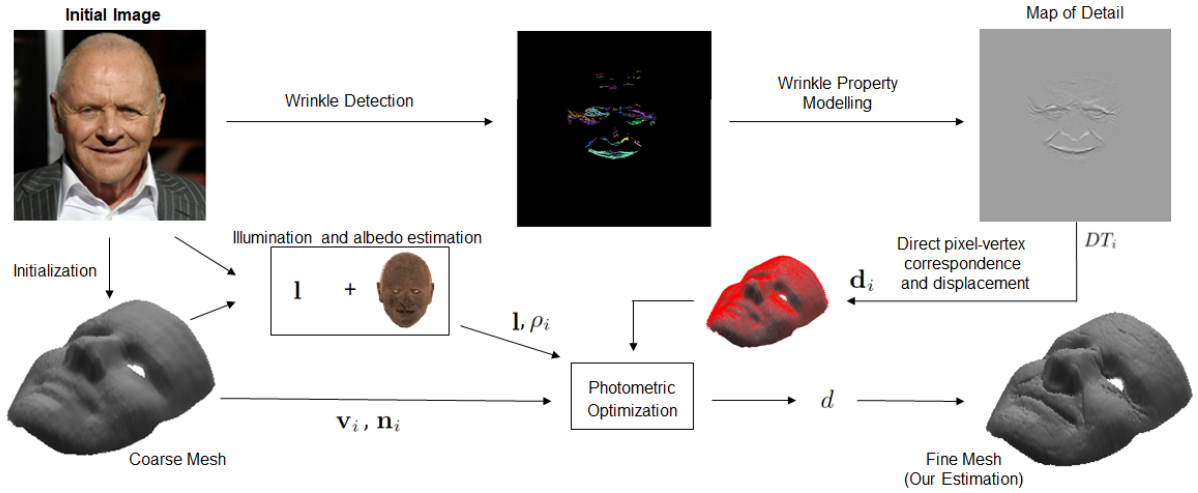


Figure 1: **An overview of our fine-detail reconstruction approach from an RGB image.** Our approach consists of two flows: 1) we use a deep-learning approach to obtain a 3D coarse mesh from the image. From this geometry can be extracted the 3D locations $\mathbf{v}_i$ and normals $\mathbf{n}_i$ for the i-th point. 2) Just considering the input image, we run a wrinkle detection algorithm based on image partial derivatives, and then group and model each of them. With this information, we can obtain a map of details DT_i to establish a direct correspondence between pixels in the image and vertices in the mesh, as well as the corresponding displacement vector $\mathbf{d}_i$. Finally, along with the illumination properties and albedo $(\mathbf{l}, \rho_i)$, we formulate the problem by means of a photometric optimization problem to recover the final detailed 3D mesh parametrized by the parameter d. As can be seen, our refining approach can recover detailed areas in 3D, in contrast to initializing approaches.

shape-from-shading techniques. In [RSOEK17], they need to train a different network to recover detail, which is extremely time and data consuming. Both of them fail to recover fine-scale detail. In [JBAT17] was presented a volumetric method as an alternative to direct fitted morphable models. However, the detail level obtained is still unsatisfactory.

Our main contribution is to develop a unified and unsupervised method that retrieves a detailed 3D mesh from a single image. It is worth noting that we do not need training data to code the wrinkle space, providing striking results even for complex shapes.

3 OUR APPROACH

In this paper, we propose a novel formulation to recover detailed 3D human faces from a single RGB image. It is worth mentioning that our approach is also capable of refining solutions given by coarse deformable face models, allowing us retrieving fine details such as wrinkles, furrows, and folds. To accomplish this goal, we define the 3D human face as an irregular and non-rigid surface that may vary due to age or facial expressions. Based on those variations, we define a parametric method that allows coding the skin wrinkles over a smooth and undetailed initialization.

To this end, we find inspiration on previous works [BKN02, LXC+15], where wrinkles were automatically modeled by means of given parameters. Nevertheless, our formulation is more general being capable of extracting from the input image all the parameters required to define the parametric model, except the depth of the furrows, which can be estimated by solving an energy minimization problem. This means no further information about the 3D face (or its corresponding projection in the image) is required. Even though it seems a simple model, due to the adaptability over different scenarios, it works properly over almost all the wrinkle types we can find in real images. An overview of our approach is displayed in Fig. 1. As we will introduce later, a map of detail is computed from the image partial derivatives, and then it is applied over the mesh. It is fast to compute even working locally on all the wrinkles, and as it can be seen in the experimental section, it provides satisfactory results outperforming current state-of-the-art techniques.

3.1 Initial 3D Face from Single Image

To obtain an initial 3D face estimation from a single image, we propose to use the convolutional volumetric regression model depicted in [JBAT17]. Later, we adjust the output to our model which works on a triangulated mesh by retrieving the surface of the volume. Since in the CNN resultant cropped image the detail is almost imperceptible, we align the mesh with the original image using the facial features of both images extracted with [ABS16]. We also estimate the scene lighting field in terms of spherical harmonics [BJ03].

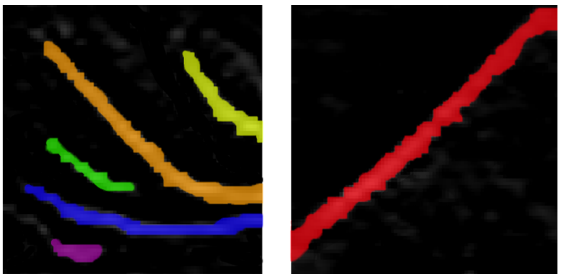


Figure 2: **Wrinkle detection.** For a specific image, we detect wrinkles across the image by clustering pixels in images partial derivatives. We obtain connected regions with an area greater than a threshold. Each wrinkle is accurately labeled and reconstructed separately since not all the wrinkles yield the same parameterization. **Left:** We observe a detection and clustering of multiple eye lines. Each color corresponds to a different wrinkle. **Right:** An example of a single nasolabial wrinkle is displayed, where small areas are ignored.

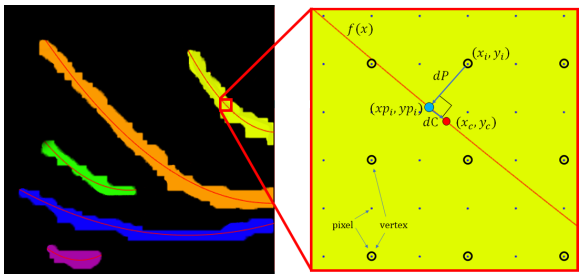


Figure 3: **Curve fitting and distance metrics. Left:** We can appreciate how the curves fit the given data. **Right:** Zooming view where we can observe the image pixel locations represented by blue dots, and these which have a corresponding mesh point by a black circle. The blue circle represents the perpendicular distance dP between a point (x_i^k, y_i^k) and $f(x)^k$, the red circle indicates the center of the wrinkle, and dC is the distance between the projected blue point (xp_i^k, yp_i^k) and the central point (x_c^k, y_c^k).

Assuming Lambertian reflectance, we can consider an initial albedo to be the mean skin color over the face. The reflection model to the i-th point can be written as $\mathbf{I}_i = \rho_i \cdot (\mathbf{l} \cdot \mathbf{H}(\mathbf{n}_i))$, where $\mathbf{I}_i$ represents the image value, ρ_i is the albedo, $\mathbf{l}$ is a 1×9 vector to represent the spherical harmonic coefficients, $\mathbf{H}$ is a 9×1 vector to indicate the spherical harmonic basis, and $\mathbf{n}_i$ is the surface normal. The process to estimate $\mathbf{l}$ and ρ_i may be iterated until the variance is sufficiently low. This model is simple and fast, but it has a limitation, does not accomplish well among cast shadows.

3.2 Wrinkle Properties

Modeling the wrinkles of a human face through a parametric formulation is a relevant area in 3D facial animation [BKN02, WWY06], with applications on face acquisition in real time [CBZB15, LWYZ17]. However, in order to produce very realistic results, these approaches normally require a large number of parameters to be known, such as the wrinkle properties as well as the material (skin) behavior. As a common practice, the location of the wrinkle furrow is represented by parametric models, such as the cubic Bezier curve [BKN02]. Unfortunately, previous formulations require these parameters to be known, or they have to be defined by the user in advance, limiting its applicability in real scenarios. To solve this limitation, we present a simple and intuitive method to retrieve the detailed furrows that better adapt over the image plane. Our method is able to extract from the image all those parameters, without the need for any training data at all. In addition, our approach can work on different mesh resolutions, and recover from large- to fine-scale details. It is suitable to describe fine-scale furrows since they can be adjusted with only three points. As a limitation, a furrow with less than three associated points cannot be retrieved.

Localization and clustering of wrinkle pixels. First of all, we set the images partial derivatives in both x- and y-directions. Both directions are used to determine regions of the image where changes in texture or geometry occur, i.e., allowing us to detect the wrinkles too. To avoid miss-detection from noisy measurements, we select the absolute value of magnitudes which are higher than a specific threshold (0.25 in our experiments). Once the initial candidates are detected, we analyze the connectivity between selected pixels, determining so different regions where all the pixels are connected (some examples are displayed in Fig. 2). We consider K sets of 8-connected pixels where each has been previously detected a wrinkle at pixel-level. We group all the pixels in a determined wrinkle k as $\mathbf{I}(x_i^k, y_i^k)$.

Curve Fitting and Distance Metrics. We now describe how these clustered points can affect to the geometric relief. Following classical curve fitting methods for data analysis, we introduce a second-order polynomial that better fits every curve. Note that for partial derivative with a major component in x, we swap axis since a well-defined function associates one, and only one, output to any particular input. So for each region k, we define a function $f(x)^k$ that better fits the data, such that:

$$f(x)^k = a^k x^2 + b^k x + c^k \,, \tag{1}$$

where the tuple (a^k, b^k, c^k) have to be estimated for every curve k. To define our parametric model over the defined furrow in a region, we propose to use the location of pixels (x_i^k, y_i^k) in the image, the curve defined by $f(x)^k$, and its corresponding centroid (x_c^k, y_c^k) such as:

$$(x_c^k, y_c^k) = \left(\frac{1}{R} \sum_i x_i^k, \frac{1}{R} \sum_i y_i^k \right) \quad (2)$$

where R represents the number of pixels in the k-th region.

Wrinkle Modeling. We next exploit the information available from the image and the previously extracted data to model a wrinkle as a furrow or relief. For every group k, let us define dP as the perpendicular distance between the point (x_i^k, y_i^k) and $f(x)^k$. In the same manner, we also introduce dC as the distance between a projected point (xp_i^k, yp_i^k) and the corresponding central point (x_c^k, y_c^k) projection on $f(x)^k$ (both distances are represented in Fig. 3).

Considering previous definitions, we now assume that the maximum influence of the depth parameter is where dP becomes null, with the global maximum at the central point (x_c^k, y_c^k) projection, then it loses influence while distances dP and dC increase.

To obtain the point (xp_i^k, yp_i^k), we first find the tangent to $f(x)^k$ at point x_i^k as $T_s = 2a^k x + b^k$. After that, we find the negative reciprocal slope, with is the normal slope $N_s = -1/T_s$, and the normal line to $f(x)^k$. Finally, we compute the two possible solutions of the second-order polynomial and choose the one with smaller distance to the point (x_i^k, y_i^k). On summary, both locations can be computed as:

$$xp_i^k = \frac{-b^k + N_s \pm \sqrt{(b^k - N_s)^2 - 4a^k(c^k + N_s x_i^k - y_i^k)}}{2a^k},$$
$$yp_i^k = a^k xp_i^2 + b^k xp_i^k + c^k.$$

We also introduce the height h of the wrinkle, and it can be defined by the maximum distance between any point (x_i^k, y_i^k) and the centroid (x_c^k, y_c^k) projection on $f(x)^k$. Additionally, it is worth noting that the width of the wrinkle we denote as w is the maximum distance that a point can have with $f(x)^k$ in the perpendicular direction.

According to previous definitions, we can model the indentation/relief of every point belonging to the region according to the parameters dP, dC, w, and h. So the effect of the furrow on the center is maximum when both dP and dC are zero and soften as the distances get larger. We now define a detail map DT which is a map with the same size as the image at full resolution and contain information about how the skin should wrinkle at each position. Its value for every i-th pixel in the region is defined as:

$$DT_i = \left(1 - \frac{dC_i}{h}\right) \cdot \left(-exp\left(\frac{-dP_i}{w}\right)\right). \quad (3)$$

Note that this map gives information on how the skin should wrinkle along with the image depending on the distances dP and dC, that provides a smooth map of simulated wrinkles. Thus, the smoothness property of the resultant detailed shape is guaranteed since the pixels that are close also have a similar value. However, meshes with a very poor resolution may need a smoothing step to adjust the result to the mesh geometry, since the resultant detail may be too sharp. Finally, we normalize DT between -1 and 1. Then, while the wrinkle sunk in the lowest values of DT, the remaining pixels on the outer part (with larger values in DT), slightly lift to produce a more realistic effect. The values not referring to any wrinkle are set as zero.

The transference from the map of detail to the mesh is done by direct pixel-vertex correspondence. Each vertex of the mesh $\mathbf{v}_i$ is displaced along its normal direction $\mathbf{n}_i$ according to the value of DT_i in order to determine the corresponding displacement $\mathbf{d}_i$ defined as:

$$\mathbf{d}_i = \mathbf{n}_i \cdot d \cdot DT_i, \quad (4)$$

where $d \cdot DT_i$ is a scalar representing the displacement magnitude in the direction of the normal vector.

3.3 Depth Estimation via Energy Minimization

In order to retrieve the 3D reconstruction from a single image, we propose to minimize a photometric loss function. As DT is normalized between -1 and 1, d determines the exact scale of the shift. To automatically find the d parameter, we minimize the following photometric energy:

$$\mathscr{E}(d) = \arg\min_d \sum_{i=1}^{I} \|\mathbf{I}_i - \rho_i \cdot (\mathbf{l} \cdot \mathbf{H}(\mathbf{n}_i(\mathbf{v}_i + \mathbf{d}_i)))\|_2^2$$
$$\text{subject to } \mathbf{d}_i = \mathbf{n}_i \cdot d \cdot DT_i \quad (5)$$

where the displacement vector $\mathbf{d}_i$, is a function of the parameter d we have to estimate. $\mathbf{n}_i(\mathbf{v}_i + \mathbf{d}_i)$ refers to the normal of the surface with the estimated displacement. This optimization is resolved using the Matlab least-squares solver.

4 EXPERIMENTAL EVALUATION

We next present quantitative and qualitative results of our method on a wide range of subjects with different gender, ethnics, age, and facial gestures. Unfortunately, we cannot directly evaluate our approach due to a one-to-one correspondence between 3D ground truth, and the corresponding estimation is not available in practice. To solve this, and to provide a quantitative evaluation, we propose to align both 3D meshes and then computing an error between closest points on the meshes by following the normal direction. The error ε_{3D} can then be defined as $\varepsilon_{3D} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{v}(i)' - \mathbf{v}_{gt}(i)\|$, where

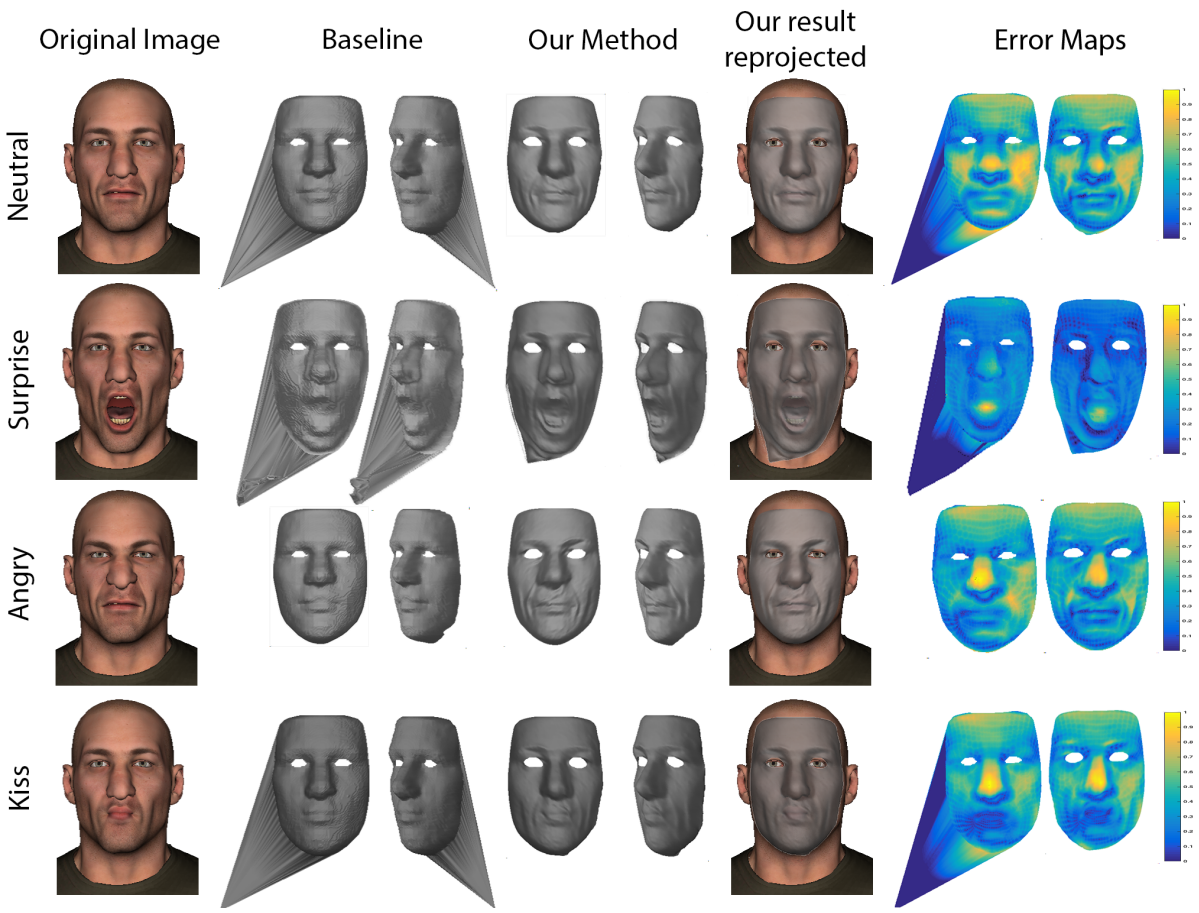


Figure 4: **Detailed 3D face reconstruction from a single image on synthetic data.** We represent a qualitative evaluation and comparison of four different facial expressions, one per row. **First column:** Rendered synthetic image we use as input. **Second and Third columns:** Frontal and side views of the 3D face reconstruction we obtain by using [JBAT17]. Particularly, we use this estimation as an initialization. **Fourth and Fifth columns:** We display the same estimations after applying our formulation, which is also reprojected over the original image in the **Sixth column**. **Seventh and Eighth columns:** A vertex error map is represented between the baseline [JBAT17] and our estimation with respect to the 3D ground truth, respectively. As can be seen, our approach can recover a larger amount of fine details in 3D. Best viewed in color.

$\mathbf{v}(i)'$ represents our 3D estimation of the i-th point and $\mathbf{v}_{gt}(i)$ is the corresponding ground truth, respectively. To compute the error, we follow the process depicted in the literature [RLMNA18] to find correspondences between aligned meshes with different elements. For further details, we refer the reader to this paper. Finally, we also establish a comparison with respect to state-of-the-art approaches.

4.1 Synthetic Images

Firstly, we evaluate our approach on synthetic data by using the Victor dataset [GVWT13], which includes several 3D meshes on a wide variety of facial gestures: pain, angry, sing, smile, kiss, sad, or surprise to name a few. This means this dataset provides plenty of medium-level wrinkles and strong cheek and nasolabial lines. It also includes tiny details around the eyes, however, it is very subtle and cannot be easily captured with the given image resolution. We propose to use this dataset to present a quantitative evaluation since the 3D ground truth is available for every mesh. As we previously introduced, to initialize our approach we obtain an initial 3D face by applying the CNN-based baseline [JBAT17]. As it was also claimed, this solution cannot recover properly the high-frequency geometry, producing smooth solutions where most of the detailed shapes are missed. However, this solution is accurate enough for initialization. Thanks to our formulation we can model every wrinkle in the shape, obtaining more detailed and realistic solutions.

We numerically evaluate our approach on eight facial expressions and establish a comparison with respect to the baseline [JBAT17]. These results are summarized in Table 1. As it is shown, our approach outperforms current solutions in terms of 3D accuracy. Unfortunately, the numerical improvement in the geometric er-

Expre.		Neutral	Sad	Angry	Pain	Surprise	Kiss	Happy	Sing	Average
[JBAT17]	$\varepsilon_{3Dtotal}$	0.022	0.040	0.039	0.037	0.064	0.050	0.055	0.059	0.046
	$\varepsilon_{3Dwrink.}$	0.808	0.811	0.814	0.814	0.613	0.820	0.811	0.813	0.788
	$\mathscr{E}(d)_{total}$	0.393	0.518	0.261	0.494	0.259	0.422	0.240	0.133	0.340
	$\mathscr{E}(d)_{wrink.}$	0.157	0.155	0.160	0.154	0.171	0.154	0.163	0.149	0.158
Ours	$\varepsilon_{3Dtotal}$	**0.021**	**0.034**	**0.036**	**0.036**	**0.042**	**0.044**	**0.051**	**0.054**	**0.039**
	$\varepsilon_{3Dwrink.}$	**0.791**	**0.797**	**0.801**	**0.801**	**0.602**	**0.801**	**0.797**	**0.799**	**0.774**
	$\mathscr{E}(d)_{total}$	0.389	0.508	0.253	0.486	0.254	0.414	0.233	0.126	0.332
	$\mathscr{E}(d)_{wrink.}$	0.125	0.123	0.127	0.128	0.151	0.122	0.134	0.136	0.131
Ours	nW	21	28	27	23	27	22	22	19	23.6
	$t(s)$	4.996	2.573	2.570	2.436	7.282	7.306	2.593	2.491	4.031

Table 1: **Quantitative evaluation and comparison on synthetic images.** The table reports the 3D error ε_{3D} together with the energy value $\mathscr{E}(d)$ in Eq. (5) for the baseline [JBAT17] and for our approach. Both metrics are computed toward the full set of points and only on the adjusted vertices, denoted by *wrink*. We also add the number of detected wrinkles nW, and the computation time $t(s)$ in seconds for our approach.

Input		Img1	Img2	Img3	Img4	Img5	Img6
[JBAT17]	$\mathscr{E}(d)_{total}$	0.2093	0.2543	2.2219	0.2969	0.3298	0.3067
	$\mathscr{E}(d)_{wrink.}$	0.1615	0.1810	0.2344	0.2109	0.3089	0.2130
Ours	$\mathscr{E}(d)_{total}$	**0.2070**	**0.2505**	**2.2103**	**0.2929**	**0.3290**	**0.2968**
	$\mathscr{E}(d)_{wrink.}$	**0.1570**	**0.1729**	**0.2285**	**0.2002**	**0.3037**	**0.2101**
Ours	nW	307	16	21	73	124	18
	$t(s)$	7.844	5.3001	5.1339	6.8080	7.6930	4.6443

Table 2: **Quantitative evaluation on real images.** The table reports the photometric energy error $\mathscr{E}(d)$ (see Eq. (5)) for the baseline [JBAT17] and for our approach toward the full set of points and uniquely over the affected vertices to properly visualize the influence of wrinkles. As in the previous analysis, we also add to our approach the number of detected wrinkles nW and the computation time $t(s)$ in seconds, respectively. Images are denoted as Img1, Img2, and Img3, and they correspond to the first column in Fig. 5. In the same manner, Img4, Img5 and Img6 correspond to the second column of the cited figure.

ror is not much striking, since the distance the vertices are shifted is short, we can observe a greater difference if we only interpret the same metric on the displaced vertices. However, the differences between both estimations can be observed in a qualitative manner. Figure 4 represents a qualitative evaluation and comparison of some evaluated facial expressions between our estimation and the baseline [JBAT17]. As can be seen, our approach is able to recover the cheek lines successfully and some other details around the facial geometry, which is not recovered properly by current methods. Particularly, it can be seen how the CNN-based solution fails to retrieve medium and fine details in some expressions (see first and third rows in Fig. 4), even producing large artifacts by placing points at one of the image corners (such as neutral, surprise and kiss expressions in the same figure). All these artifacts can be solved by our approach in an efficient and effective manner.

We also provide in Table 1 the number of wrinkles detected by our algorithm. We consider this information can also be an indirect metric to evaluate the performance of our approach since a priori a higher number of detected wrinkles should provide more detailed surfaces. However, this number should not be too high in order to avoid false wrinkles due to noisy measurements. Eventually, we include the time budget on a standard desktop computer Intel(R) Xenon(R) CPU ES-1620 v3 at 3.506GHz to solve wrinkle detection, modeling, and estimation. When the CPU permits a parallel computation, the wrinkles are estimated so, and its number does not affect drastically to the total computation budget. On balance, it is worth mentioning that all of this took just a few seconds on a commodity laptop (see the last row in Tables 1 and 2).

4.2 Real Images

We next evaluate our approach on several real images. To prove the generality of our approach we propose to process a set of images with a wide range of geometries, including subjects with different gender, age, ethnic group, and facial gesture. Since no ground truth is available for these images, we provide qualitative evaluation and comparison with the baseline [JBAT17]. For completeness, we also provide the energy value obtained by using both formulations on Table 2. We can notice the same error toward the overall vertices and exclusively on the displaced vertices. As we can appreciate, the relation between the two values establishes

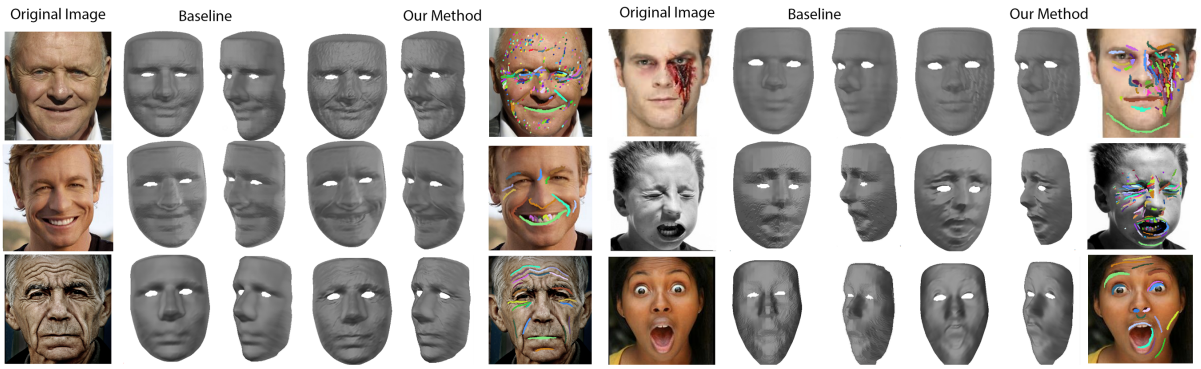


Figure 5: **Detailed 3D face reconstruction from a single real image**. Six different scenarios –varying age, gender, ethnic group, and facial expression– are displayed in rows. We represent two blocks, with the same information. **First column:** Input image. **Second and third column:** Camera and side views of the 3D reconstruction obtained by [JBAT17]. **Fourth and fifth column:** Camera and side views of our 3D reconstruction. **Sixth column:** Detected wrinkles. Best viewed in color.

Figure 6: **Detailed 3D reconstruction on real images**. Left and right display the same. **First row:** Four input images at different expressions. **From second to fourth row:** Reprojected mesh of the 3D reconstruction using [GVWT13], [JBAT17] and ours, respectively. Note that the solution provided by [GVWT13] includes a 200k-point, instead of using 20k points like us.

an indirect metric to measure the number of detections. For instance, in Img3 not all the wrinkles were properly captured due to noise (caused by filters added to the image for aesthetic purposes). The difference between values is high in comparison with other images. We can see how our method outperforms [JBAT17] toward all the metrics. Some examples from different viewpoints are displayed in Fig. 5. Again, it is worth pointing out that our method can obtain more accurate solutions than state of the art in terms of geometric details, as it can be observed in the before mentioned figure. Different types of wrinkles, as well as scars or blood marks, are satisfactorily recovered under different statuses (uncontrolled illumination, different resolution, and noise).

Next, we use four indoor images for two subjects taken from [GVWT13]. As before, we can only present a

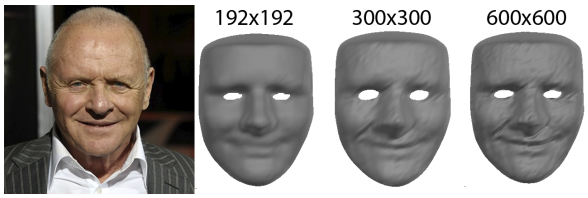


Figure 7: **3D reconstruction as a function of the image resolution.** To show the impact of the image resolution over the final result, we run our approach on different down-sampled images. While most of the details are not detected in low-resolution images, they become more accurately detected as the resolution in pictures increases.

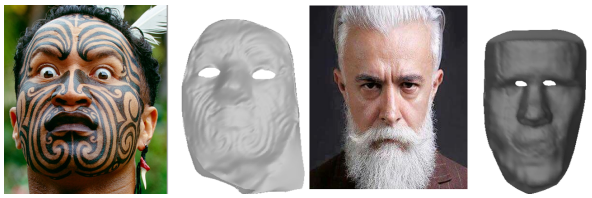


Figure 8: **Failure cases**. Two ambiguous examples our approach cannot solve properly. **Left:** A dark tattooed Maori where our algorithm fails in differentiating texture and shadow areas. **Right:** A big beard produces a self-occlusion in the face. Although the estimation is visually correct, it does not consider nor recover the human beard.

qualitative evaluation and comparison, including this time the results provided by [GVWT13]. We display the corresponding 3D reconstructions on Fig 6. Despite other techniques exploiting temporal priors and a higher resolution [GVWT13], our approach still seems to provide more detailed solutions than the remainder of the evaluated methods, even when reconstructing significantly fewer points.

In general terms, real images include more sophisticated details compared to synthetic data. Note that the design of realistic wrinkles is still a challenging problem in 3D modeling and animation. Therefore, the acquisition of realistic models from vision is essential to geometrically analyze these local details. In this context, the input image resolution is a key factor to obtain good results. As it is represented in Fig. 7, we observe a more fine acquisition when the image resolution increases.

Failure Cases. Since our formulation relies on image partial derivatives to detect wrinkles, some conditions such as shadows or texture-varying areas could produce ambiguous situations. For instance, obscure tattoos, strong cast shadows, and significant occlusions are some examples in which recovering detail becomes a hard task, and our method fails (observe some examples in Fig. 8). In the case of facial tattoos, or strong cast shadows, the algorithm proceeds to reconstruct the wrinkles where the geometry is not varying.

5 CONCLUSION

In this paper, we have proposed an intuitive and effective approach to retrieve detailed 3D reconstruction of faces from a single RGB image. Considering only the input image, our approach can obtain several features to parametrize the wrinkles without having been observed previously. This scheme allows us to model even person-specific attributes, such as scars or several shapes as a consequence of aging. Additionally, our approach is efficient since only need few seconds to solve the problem, by sorting out a photometric optimization problem. We have extensively evaluated our approach on both synthetic and real images, considering a wide range of variability in which we have outperformed existing state-of-the-art solutions. An interesting avenue for future research is to extend our formulation to handle more severe occlusions as well as a validation in real time at frame-rate.

Acknowledgments: This work has been partially supported by the Spanish Ministry of Science and Innovation under projects FireDMMI TIN2014-56919-C3-2-R, BOSSS TIN2017-89723-P, and HuMoUR TIN2017-90086-R; by the CSIC project R3OBJ 201850I099, and by the Spanish State Research Agency through the María de Maeztu Seal of Excellence to IRI MDM-2016-0656.

6 REFERENCES

[ABS16] B. Amos, L. Bartosz, and M. Satyanarayanan. OpenFace: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, 2016.

[AMAC17] A. Agudo, J. M. M. Montiel, L. Agapito, and B. Calvo. Modal space: A physics-based model for sequential estimation of time-varying shape from monocular video. *JMIV*, 57(1):75–98, 2017.

[AMCMN16] A. Agudo, J. M. M. Montiel, B. Calvo, and F. Moreno-Noguer. Mode-shape interpretation: Re-thinking modal space for recovering deformable shapes. In *WACV*, 2016.

[AMN17] A. Agudo and F. Moreno-Noguer. Combining local-physical and global-statistical models for sequential deformable shape from motion. *IJCV*, 122(2):371–387, 2017.

[AMN18] A. Agudo and F. Moreno-Noguer. Force-based representation for non-rigid shape and elastic model estimation. *TPAMI*, 40(9):2137–2150, 2018.

[AMN19] A. Agudo and F. Moreno-Noguer. Shape basis interpretation for monocular deformable 3D reconstruction. *TMM*, 21(4):821–834, 2019.

[BJ03] R. Basri and D. W. Jacobs. Lambertian reflectance and linear subspaces. *TPAMI*, 25(2):218–233, 2003.

[BKN02] Y. Bando, T. Kuratate, and T. Nishita. A simple method for modeling wrinkles on human skin. In *CG&A*, 2002.

[BV99] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH*, 1999.

[CBZB15] C. Cao, D. Bradley, K. Zhou, and T. Beeler. Real-time high-fidelity facial performance capture. *TOG*, 34(4):46, 2015.

[CCC+18] Xuan Cao, Zhang Chen, Anpei Chen, Xin Chen, Shiying Li, and Jingyi Yu. Sparse photometric 3D face reconstruction guided by morphable models. In *CVPR*, 2018.

[CCL18] N. Chinaev, A. Chigorin, and I. Laptev. Mobileface: 3D face reconstruction with efficient cnn regression. In *ECCV*, 2018.

[CWZ+14] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3D facial expression database for visual computing. *TVCG*, 20(3):413–425, 2014.

[GFT+11] A. Ghosh, G. Fyffe, B. Tunwattanapong, J. Busch, X. Yu, and P. Debevec. Multiview face capture using polarized spherical gradient illumination. *TOG*, 30(6), 2011.

[GMMB00] E. Guillou, D. Meneveaux, E. Maisel, and K. Bouatouch. Using vanishing points for camera calibration and coarse 3D reconstruction from a single image. *VC*, 16(7):396–410, 2000.

[GSSM15] P. F. U. Gotardo, T. Simon, Y. Sheikh, and I. Matthews. Photogeometric scene flow for high-detail dynamic 3D reconstruction. In *ICCV*, 2015.

[GVWT13] P. Garrido, L. Valgaerts, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *TOG*, 32(6):158–1, 2013.

[GZC+16] P. Garrido, M. Zollhöfer, D. Casas, L. Valgaerts, K. Varanasi, P. Pérez, and C. Theobalt. Reconstruction of personalized 3D face rigs from monocular video. *TOG*, 35(3):28, 2016.

[HHT+16] P. Huber, G. Hu, R. Tena, P. Mortazavian, P. Koppen, W. J. Christmas, M. Ratsch, and J. Kittler. A multiresolution 3D morphable face model and fitting framework. In *VISIGRAPP*, 2016.

[JBAT17] A. S. Jackson, A. Bulat, V. Argyriou, and G. Tzimiropoulos. Large pose 3D face reconstruction from a single image via direct volumetric CNN regression. In *ICCV*, 2017.

[JZD+18] L. Jiang, J. Zhang, B. Deng, H. Li, and L. Liu. 3D face reconstruction with geometry details from a single image. *TIP*, 27(10):4756–4770, 2018.

[LWYZ17] S. Liu, Z. Wang, X. Yang, and J. Zhang. Realtime dynamic 3D facial reconstruction for monocular video in-the-wild. In *CVPR*, 2017.

[LXC+15] J. Li, W. Xu, Z. Cheng, K. Xu, and R. Klein. Lightweight wrinkle synthesis for 3D facial modeling and animation. *Computer-Aided Design*, 58:117–122, 2015.

[LZZL16] F. Liu, D. Zeng, Q. Zhao, and X. Liu. Joint face alignment and 3D face reconstruction. In *ECCV*, 2016.

[RLMNA18] G. Rotger, F. Lumbreras, F. Moreno-Noguer, and A. Agudo. 2D-to-3D facial expression transfer. In *ICPR*, 2018.

[RSK16] E. Richardson, M. Sela, and R. Kimmel. 3D face reconstruction by learning from synthetic data. In *3DV*, 2016.

[RSOEK17] E. Richardson, M. Sela, R. Or-El, and R. Kimmel. Learning detailed face reconstruction from a single image. In *CVPR*, 2017.

[RV05] S. Romdhani and T. Vetter. Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In *CVPR*, 2005.

[SKSS14] S. Suwajanakorn, I. Kemelmacher-Shlizerman, and S. M. Seitz. Total moving face reconstruction. In *ECCV*, 2014.

[SWTC14] F. Shi, H.T. Wu, X. Tong, and J. Chai. Automatic acquisition of high-fidelity facial performances using monocular videos. *TOG*, 33(6):222, 2014.

[TTHM+18] A. Tuan Tran, T. Hassner, I. Masi, E. Paz, Y. Nirkin, and G. Medioni. Extreme 3D face reconstruction: Seeing through occlusions. In *CVPR*, 2018.

[VWB+12] L. Valgaerts, C. Wu, A. Bruhn, H.P. Seidel, and C. Theobalt. Lightweight binocular facial performance capture under uncontrolled lighting. *TOG*, 31(6):187–1, 2012.

[WBS01] M. Wilczkowiak, E. Boyer, and P. Sturm. Camera calibration and 3D reconstruction from single images using parallelepipeds. In *ICCV*, 2001.

[WWY06] Y. Wang, C.C.L. Wang, and M.M.F. Yuen. Fast energy-based surface wrinkle modeling. *Computers & Graphics*, 30(1):111–125, 2006.

[ZNI+14] M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, et al. Real-time non-rigid reconstruction using an RGB-D camera. *TOG*, 33(4):156, 2014.

[ZTG+18] M. Zollhöfer, J. Thies, P. Garrido, D. Bradley, T. Beeler, P. Pérez, M. Stamminger, M. Nießner, and C. Theobalt. State of the Art on Monocular 3D Face Reconstruction, Tracking, and Applications. *Computer Graphics Forum*, 37(2), 2018.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Coffee Grading with Convolutional Neural Networks using Small Datasets with High Variance

Serawork Wallelign[1,2], Mihai Polceanu[1,*], Towfik Jemal[2], Cédric Buche[1]
[1]LAB_STICC, ENIB, CNRS, France
[2]JiT, Jimma University, Ethiopia
[wallelign, polceanu, buche]@enib.fr, towfikjemal@yahoo.com

ABSTRACT

Convolutional Neural Networks (CNNs) have been established as a powerful class of models for image recognition problems. Despite their success in other areas, CNNs have been applied only for very limited agricultural applications due to the need for large datasets. The aim of this research is to design a robust CNN model that classifies raw coffee beans into their 12 quality grades using small datasets which have high data variability. The dataset contains images of raw coffee beans acquired in two sets using different acquisition technique under varying illuminations which poses a complex challenge to designing a robust model. To design the model, preprocessing techniques were applied to the input in order to reduce task irrelevant features. But adding the preprocessing techniques did not improve the performance of the CNN model for our dataset. We have also used ensemble methods to solve the high variance that exists in networks when working with small datasets. Finally, we were able to design a model that classifies the beans into their quality grades with an accuracy of **89.01%** on the test dataset.

Keywords

CNN, deep learning, ensemble methods, small dataset.

1 INTRODUCTION

Deep learning is making major advances in solving problems that have resisted the best attempts of the artificial intelligence community for many years. It has shown outstanding performance and is applied in many areas like business, science and government [LBH15]. As deep learning has been successfully applied in various domains, it has recently also entered the domain of agriculture. In particular, Convolutional Neural Network (CNN) has been applied for recognition and classification tasks in agricultural applications [FPF11] [RCC15]. However, a considerable barrier in the use of CNN is the need for large datasets, which would serve as the input during the training procedure. In spite of data augmentation techniques, which can enhance datasets with label-preserving transformations, in reality a dataset containing several hundreds of images may often prove to be insufficient, depending on the complexity of the problem under study [KPB18]. In the domain of agriculture, only few publicly available datasets (eg. PlantVillage[HS+15], LifeCLEF[JMG+14], MalayaKew[LCWR15] and UC Merced[YN10]) exist for researchers to work with, and in many cases, researchers need to develop their own sets of images manually. The images are usually collected at multiple sites using different acquisition techniques under varying illuminations which poses a complex challenge to designing a robust model.

The fact that CNNs can learn important features from the raw data minimizes the need to use image processing techniques especially when enough data is available. The most commonly used preprocessing techniques are mean normalization, i.e, subtracting the mean of the dataset from each image, and scaling (limiting the dataset to be in the same range) the input before feeding to the network. This is important to speedup learning as well as increase the generalization ability of the network when the dataset contains enough images to represent different acquisition variations. However, this may not be enough when working with small datasets which have high data variability. We will investigate the effect of applying additional preprocessing on the performance of a CNN model when working with small dataset with high data variability.

Coffee is one of the most important globally traded agricultural commodities, with consumption occurring mostly in developed countries and production in devel-

* Current affiliation: University of Greenwich, UK

oping ones. It is one of the most important agriculture activities to earn foreign exchange in many tropical and subtropical countries [Ayi14] [FPF11]. The quality of produced coffee depends on the way it is cultivated and processed, which in turn determines its price on the market. Therefore, all produced coffee has to go through the grading process before it is introduced into the market.

Grading is the process of categorizing coffee beans on the basis of various criteria such as size of the bean, where and at what altitude it was grown, how it was prepared and picked, and how good it tastes [KJM+15]. There is no single agreed method of coffee grading among the top producing countries of the world. For example, countries like Indonesia base grading on the amount and type of defects present in the beans [FPF11], while other countries like Ethiopia use the physical characteristics (size, color, shape and uniformity of the beans), defects, presence of foreign objects as well as cup taste.

Despite the fact that coffee is an important cash crop, grading of coffee is performed manually. This method employs visual and manual methods of inspection exposing the quality assessment to inconsistent results and subjectivity. As a result, supporting the human operator systems with the consistent, non-destructive and cost effective automated system of coffee quality classification and determination is necessary for such commercial products that generate an important amount of income.

To the best of our knowledge, only [Ayi14], [TAG13], [FPF11] and [dOLB+16] applied computer vision to detect defective coffee beans, classify beans according to their geographic origin and grade the raw beans based defect systems respectively. Although research in computer vision has advanced in recent years, only image processing methods and traditional artificial neural networks were used. This limits the performance of the model to depend heavily on the underlining image processing and feature extracted methods used. However, deep learning techniques, which can automate the feature design step, are outperforming traditional, handcrafted features based methods. In this study, modern methods to train CNNs are used to design a system that grades coffee using images of raw coffee beans. The images are collected in two sets where they differ in resolution, scale and imaging protocol used. We will also investigate how these variations affect the training of CNN. Our contribution is to automate the coffee grading process by designing a robust CNN model that classifies coffee beans into their quality grades from images of raw coffee beans. We were able to design a strong classifier from several weak models by applying ensemble methods. Though color correction preprocessing and augmentation methods were applied to decrease task irrelevant features, they did not improve the performance of the model.

The rest the paper is organized as follows: section 2 discusses the manual grading of raw coffee beans; Section 3 deals with data collection and preparation; Section 4 describes the methodology used to design the model. The experimental results obtained are presented in section 5 and we conclude by recommending methods for future improvement.

2 MANUAL GRADING OF RAW COFFEE BEANS

The current practice of coffee grading is conducted manually based on a classification into four basic groups by type and market: washed beans for the domestic and international markets, and unwashed beans for each of these markets (Figure 1). Different criteria are used for each group to grade the beans. For example, the presence of beans covered by husk (refer to Figure 2c) is a criterion for unwashed categories but not for the washed categories. The experts (cuppers) assign scores for the raw coffee beans through one or a combination of two methods: Raw Quality and Cup Value (Cup Test).

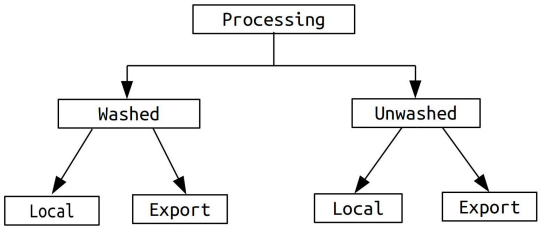


Figure 1: Grading categories based on way of processing and market. Experts use different criteria for each of the four categories

Figure 2: Some defects that affect the grade of coffee a) broken beans b) beans damaged by insects c) beans covered by husk d) unmatured beans e) foreign objects f) unmatured and broken

The raw quality is performed by examining the physical properties of the raw coffee beans. It entails characteristics like defects, shape and make, color and odor. The

cup value is determined by roasting and tasting sample coffee for acidity, cleanness and flavour. The final score is the sum of the raw value and cup value. The grade is then decided based on discretizing the score using intervals for each grade.

For each coffee shipment, a 3kg representative sample is taken from a 10-15 ton delivery. The sample is then coded and subjected to size screening and moisture tests. If it passes these tests, out of the 3kg, 300g is used for raw evaluation. The remaining sample is divided for roasting, future reference and client's display. The sample for raw evaluation is sorted manually by hand picking to separate each defect. A score is then given based on the ratio of defects to the entire sample or using the number of defective beans present. Some of the defects are shown in Figure 2. This accounts for 20% of the total score. Then each cupper works individually on each sample and records their opinion for every criteria. After every expert finished examining the sample, the final value of each attribute is decided by discussion among the experts. If there is a difference on the points, they will convince one another and elaborate their reasons to adjust their scores in order to reach a consensus. The final score will be given by an overall agreement of all the experts. The score is converted to the grade value based on in which range it lies. For example, coffee beans whose score is in the range 85 and above will be classified in grade-1 while beans with a score in the range 75 and 84 will be classified as Grade-2 coffee.

3 DATA COLLECTION AND PREPARATION

The images of coffee bean samples were collected at Jimma grading center in Ethiopia. The beans from the 300 gram sample used for raw evaluation were used to prepare the dataset. A small portion is taken from the sample, dispersed evenly in an A4 white paper then a picture is captured after which the beans are placed in a separate container. The process is repeated until the picture of all the beans in the sample have been taken. Capturing the images this way prevents the beans from overlapping and results in a sample image that is representative of the whole container.

The images were collected in two rounds nine months apart using different capturing devices. The first set of images was collected during the harvest season resulting in a dataset dominated by samples from the washed category. Whereas, the second set was collected when the beans from the previous year's harvest were graded which results in more samples from the unwashed category.

In the first round 1266 images of coffee beans from 12 grades with resolution of 2268x4032 pixels were captured using Samsung s7 edge camera (Figure 3). This

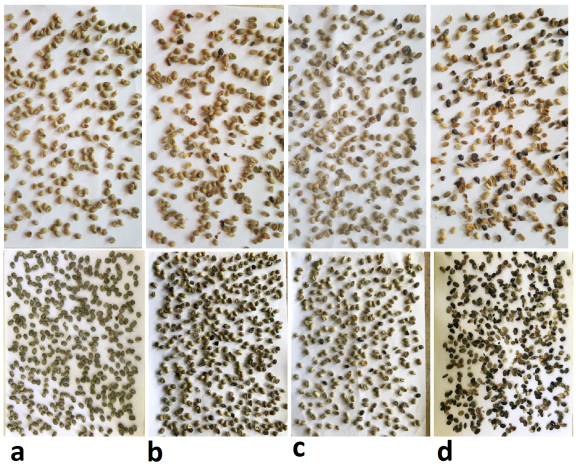


Figure 3: Sample images from the dataset. Top row: images collected in the first round. Bottom row: images collected in the second round. a) Grade-2 b) Grade-7 c) Local-1 d) Local-5A

set is mainly populated by samples from washed category (G1, G2 and G3). In the second round, 843 images with resolution of 3024x4032 pixels were captured by iPhone 7 plus camera dominated by samples from unwashed category (G6, G7). When trying to fit all the beans on the A4 paper, we were indirectly controlling the distance from the device to the beans. This created a scale difference between the two sets since the cameras have different resolution. There was no control on lighting while capturing the images. Table 1 shows the number of samples per grade collected for the two datasets. Since the collected images have high resolution, they are cropped into two along the longer side to increase the number of samples without losing important information.

Grade	Dataset1	Dataset2	Total
Grade-1	115	0	115
Grade-2	614	75	689
Grade-3	84	29	113
Grade-6	56	318	374
Grade-7	116	323	439
Grade-8	43	13	56
Local-1	14	16	30
Local-2	156	19	175
Local-3	30	11	41
Local-4	15	14	29
Local-5A	16	10	26
Local-5C	7	15	22
Total	**1266**	**843**	**2109**

Table 1: The number of samples per class in each dataset. Almost half of Dataset1 is from the washed catagory while about 76% Dataset2 is from unwashed category.

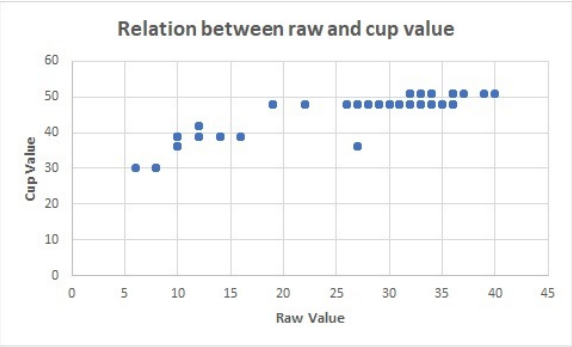


Figure 4: Correlation between the raw value and cup value

During the on-site survey at the grading organization, the experts stated that the physical appearance of the beans has an effect on the cup value since the presence of defects and foreign bodies affect the taste. To test the hypothesis, the correlation between the raw value and cup value of the sample data is calculated. The plot of the raw value versus the cup value is shown in Figure 4.

The correlation coefficient between the raw value and the cup value is 0.87 which indicates the existence of a strong relationship between the two values. Therefore, only considering the physical properties of the beans should be sufficient to obtain a model of the coffee grading process.

4 METHODOLOGY

This section discuses the steps followed and methods applied during the model design.

4.1 CNN architecture

Since we have a small dataset, the main focus during the design of the CNN architecture was to obtain a simple model with few trainable parameters. To achieve this goal, we followed two methods. First, to use transfer learning on existing trained model. It is common practice that features extracted by training a CNN in a fully supervised manner in large-scale object recognition datasets be reused for a novel generic task [DJV+14]. Therefore, we applied transfer learning on the pre-trained VGG[SZ14] model by removing the last fully connected layers. Second, to design a new architecture and compare the result with the one obtained from transfer learning.

The architecture of the new CNN model is adapted from the same model that we used for transfer learning except two modifications. Instead of using fully connected layer for the classifier, we used a 1x1 conv layer followed by global average pooling. It is stated that if the image area covered by units in the top most convolutional layer covers a portion of the image large enough to recognize its content then fully connected layers can also be replaced by simple 1-by-1 convolutions [SDBR14]. This has a great advantage when working with small datasets by regularizing the network further since it results in a smaller number of model parameters than the fully connected layer. The second modification is, although using filters smaller than 5x5 pixels is recommended for use in the first convolutional layer, a 7x7 filter with stride=3 worked better for our dataset.

For the sub-sampling layer we used overlapping max-pooling with stride = 2 and kernel size = 3. For activation layers, using LeakyRelu (alpha = 0.001) resulted in relatively faster training and a more stable network than with the ReLU activation function. Therefore LeakyRelu is used after every convolutional layer except the last one where softmax activation is used to generate the probability of a sample being in each grade. The architecture of the designed model (referred to hereafter as CNN_1) is summarized in Table 2.

Layer	Filters	Size, stride	Remark
Input		224x224	RGB
Conv1	96	7x7,3	LeakyReLU
Conv2	96	1x1,1	LeakyReLU
Conv3	192	3x3,1	LeakyReLU
Conv4	192	1x1,1	LeakyReLU
MaxPooling	192	3x3,2	
Conv5	384	3x3,1	LeakyReLU
Conv6	384	1x1,1	LeakyReLU
MaxPooling	384	3x3,2	0.5 dropout
Conv7	950	1x1,1	LeakyReLU
Conv8	12	1x1,1	12 outputs
GlobalAverage			
Softmax			

Table 2: Architecture of the model

4.2 Dataset variations

To see how the data collection method affects the model's generalization ability, we designed and trained a simple CNN model using only images from Dataset1. The model classifies the beans with an accuracy of 94.58%. But when it is tested using the images from Dataset2, the accuracy dropped to 8%. Though different image augmentation techniques were applied during the training process to cover the possible variations in the unseen data, the model fails to detect important features to classify beans correctly in the second set. The scatter plot (Figure 5) of the datasets shows that the difference in data distribution between them. This clearly shows that how the images were captured, camera differences, illumination conditions and scale introduced task irrelevant features to datasets. The data mismatch

problem can be addressed by applying preprocessing techniques to create similarity between the datasets and adding artificial synthesized data during the training to increase the model's generalization ability. In addition to another geometric augmentation techniques, random scaling is used to make the model robust to scale changes in the data. Color correction augmentation and preprocessing methods were also added to solve the mismatch because of the camera differences and lighting conditions. These techniques are discussed in the next section.

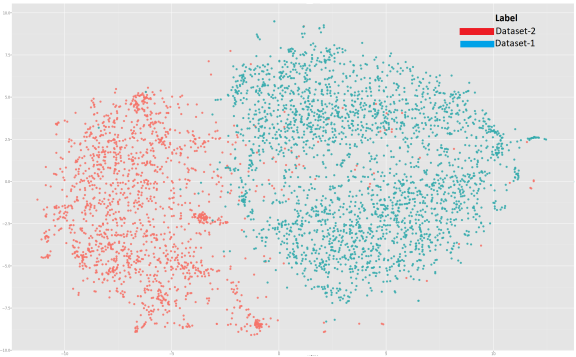


Figure 5: t-SNE plot of the distribution of the images in the datasets after feature reduction is performed using PCA. blue-Dataset1, red-Dataset2

4.3 Augmentation for color correction

The colors that are present in images are determined by the intrinsic properties of objects and surfaces as well as the illuminant of the scene [GGVDW11]. For a robust color-based system, these effects of the light source should be filtered out. This effect, i.e the ability to account for the color of illuminants, is known as color constancy. Many computer vision problems in both still images and videos can make use of color constancy processing as a preprocessing step to make sure that the recorded color of the objects in the scene does not change under different illumination conditions. But most often in deep learning models only intensity normalization is performed on the input images (VGG [SZ14], Inception [SLJ+15]). This may be enough when working with large datasets with millions of images but not for small datasets which lack in representation. Therefore, we applied a white balancing technique to preprocess the input images for color constancy and compare the result with intensity normalization. Gray-World[Buc80] color constancy technique is used. It is based on the assumption that the color in each sensor channel averages to gray over the entire image. We chose this algorithm because of its simplicity and fast computation time.

Gamma correction is another operation that digital cameras perform in order to match the human perception of an image. In digital cameras the received

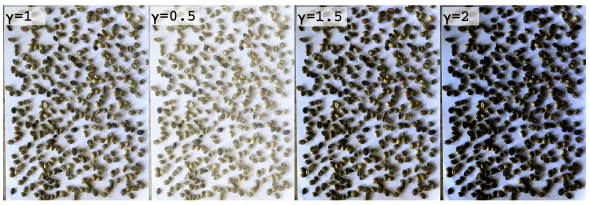


Figure 6: Sample gamma corrected images. Images appear darker as we increase the value of gamma. From left to right gamma = 1(no change), gamma = 0.5, gamma = 1.5 and gamma = 2.

signal from the sensors is linearly proportional to the light source that hits the sensor. To account for the non linearity of this process in humans, a non linear transformation function is applied. Sample gamma transformed images are shown in Figure 6. Images appear darker as the value for higher values of gamma. However this transformation is dependent on characteristics of the display and on the imaging device manufacturer. Therefore we added a gamma correction augmentation technique during the training of the model.

5 RESULTS

The models were implemented in Keras deep learning framework [C+15] on a single GTX 1070 GPU. The dataset is divided into three sets for training (70%), validation (10%) and testing (20%). All images were resized to 224x224 pixels, intensity values were scaled between 0 and 1. To compare the preprocessing techniques, two version of datasets were prepared. In one, let's call it DS1, images were normalized by subtracting the mean of the training set from each image. The second, DS2, was prepared using the white balancing color constancy technique. The experiment was conducted on both datasets DS1 and DS2. Each model is trained for 1000 epochs with early stopping, Adam optimization (learning rate = 0.0001) is used with categorical cross entropy loss function.

For the transfer learning, after removing the fully-connected layers of the VGG, two layers with 1024 nodes and 50% dropout were added followed by an output layer with 12 nodes equivalent to the number of grades. Then we trained only the new fully connected layers by keeping the weights in the other CNN layers the same as the one learned using the Imagenet dataset [DDS+09]. Neural networks are sensitive to initial conditions, both in terms of the initial random weights and in terms of the statistical noise in the training dataset. This stochastic nature of the learning algorithm means that each time a neural network model is trained, it may learn a slightly different version of the mapping function from inputs to outputs, that in turn will have different performance on the training and holdout datasets. Therefore, we trained the models

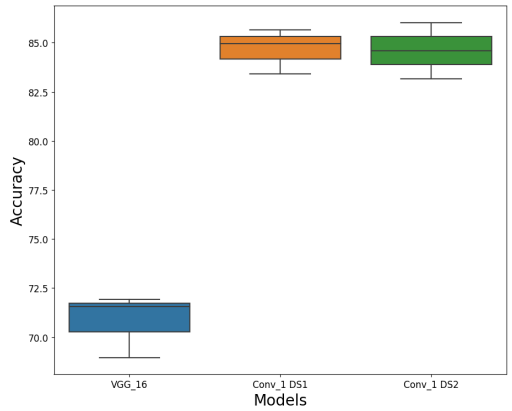


Figure 7: Performance of models for DS1 and DS2. There is high variance on test performance of the models when trained using different initial conditions

several times starting from random initial conditions. The performances of the models on DS1 and DS2 for different runs is shown in Figure 7. It can be observed from the plot that training the models starting from different initial conditions resulted on varying prediction performance making optimization and choosing the final model very challenging. We can see from the result VGG-16 has lower performance than CNN_1 for this dataset. It was also experimented retraining the last two convolutional layers but the performance did not show much improvement compared to the CNN_1.

A solution for high variance in networks, while also improving accuracy, is using ensemble learning [Die00]. Ensemble learning is training several models varying the initial conditions and then combining the predictions from each model to make one final prediction. We applied two types of ensemble methods, Checkpoint Smoothers (CS) and Checkpoint Ensemble (CE)[CLL17], to compare the performance between the models as well as to choose the final model. CS averages the weight of n models around the model with minimum validation (MV) loss while CE averages the predictions of the models. We picked the best performing models based on MV loss when trained with both datasets (models found at run-3 for DS1 and at run-2 for DS2) and applied CS and CE for $n = 10$, $n = 20$ and $n = 50$. The test accuracy of the models for different values of n is summarized in Table 3.

As it can be seen from the table, applying color correction preprocessing and augmentation did not bring any improvement on the performance of the model. Also using ensemble methods results in better prediction performance than the model selected based on minimum validation loss. Both the CE and CS models have comparable results. We chose the model designed using CS since it takes less prediction time than CE with equal

	CNN_1 (DS1) %	**CNN_1 (DS2) %**	Remark
run-1	84.95	84.25	MV
run-2	85.66	**86.02**	MV
run-3	**85.78**	85.07	MV
$n = 10$	88.40	87.68	CE
	88.39	87.80	CS
$n = 20$	87.80	87.20	CE
	88.51	87.68	CS
$n = 50$	88.51	87.80	CE
	89.1	**88.34**	CS

Table 3: Summary of the experimental results. The model's prediction performance is better when the dataset is preproccessed using only mean normalization.

number of models. Finally, the CS ensemble model, for $n = 50$, trained using DS1 is selected as a final model.

The Top-2 accuracy of the selected model is 98.22%. Also the confusion matrix (Figure 8) of this model on the test dataset shows most of the misclassifications occur between two neighbouring grade values. This maybe due to the fact that in the manual classification the grade value is inferred based on which range the final score lies, i.e for example if the score lies in the range 75 to 84, it will be a Grade-2 coffee and if it lies between 63 and 74 then its grade will be Grade-3. Therefore, two coffee beans with a total score of 75 and 74 will be classified in Grade-2 and Grade-3 respectively even though there is no significant difference in their appearances.

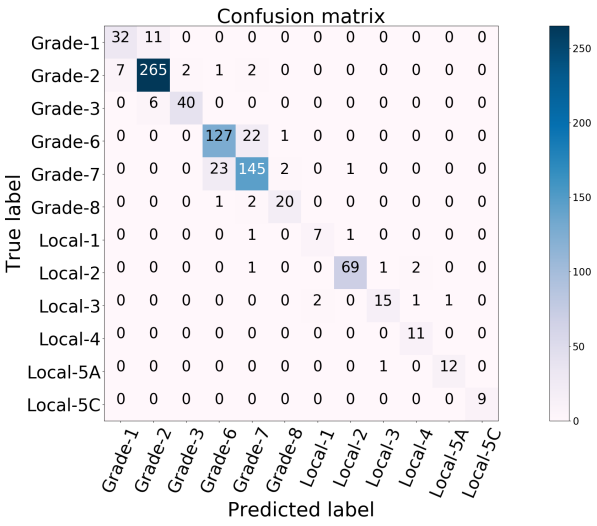


Figure 8: Confusion matrix of the model

6 CONCLUSIONS

In this work we designed a CNN model to classify raw coffee beans into 12 quality grades and achieved 89.1% on the test dataset. It was also shown that the CNN

model can extract features and classify the input images with minimal preprocessing. White balancing the images and adding color correction to the augmentation function did not improve the performance of the model. We have also implemented ensemble methods to design a strong model from several weak-classifiers. In the error analysis, we have seen that most misclassifications occur between two neighbour classes. In the future we will investigate the effect that treating the output as a classification problem rather than a regression problem had on the performance of the model.

During our stay in the grading organization, we were informed that the experts estimate their repeatability between 80 - 85%. Since the model was inferred based on the labels given by the experts, its prediction performance is affected by this error. We did not use any method to address wrong labels due to human errors. In the future, we will investigate using unsupervised or semi-supervised techniques to decrease the effect of noise present due to labelling.

7 ACKNOWLEDGMENTS

This research is funded by the French Embassy and the Ethiopian Ministry of Science and Higher Education(MoSHE) through the higher education capacity building program.

8 REFERENCES

[Ayi14] Betelihem Mesfin Ayitenfsu. Method of coffee bean defect detection. *International Journal of Engineering Research & Technology*, 3:2355–57, 2014.

[Buc80] Gershon Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin institute*, 310(1):1–26, 1980.

[C+15] François Chollet et al. Keras, 2015.

[CLL17] Hugh Chen, Scott Lundberg, and Su-In Lee. Checkpoint ensembles: Ensemble methods from a single training process. *arXiv preprint arXiv:1710.03282*, 2017.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

[Die00] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[DJV+14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[dOLB+16] Emanuelle Morais de Oliveira, Dimas Samid Leme, Bruno Henrique Groenner Barbosa, Mirian Pereira Rodarte, and Rosemary Gualberto Fonseca Alvarenga Pereira. A computer vision system for coffee beans classification based on computational intelligence techniques. *Journal of Food Engineering*, 171:22–27, 2016.

[FPF11] Faridah Faridah, Gea OF Parikesit, and Ferdiansjah Ferdiansjah. Coffee bean grade determination based on image parameter. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 9(3):547–554, 2011.

[GGVDW11] Arjan Gijsenij, Theo Gevers, and Joost Van De Weijer. Computational color constancy: Survey and experiments. *IEEE Transactions on Image Processing*, 20(9):2475–2489, 2011.

[HS+15] David Hughes, Marcel Salathé, et al. An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv:1511.08060*, 2015.

[JMG+14] Alexis Joly, Henning Müller, Hervé Goëau, Hervé Glotin, Concetto Spampinato, Andreas Rauber, Pierre Bonnet, Willem-Pier Vellinga, Robert B Fisher, and Robert Planquè. Lifeclef: Multimedia life species identification. In *EMR@ ICMR*, pages 7–13, 2014.

[KJM+15] Dae-Joong Kwon, Hee-Jeong Jeong, Hyeyoung Moon, Hak-Nam Kim, Jee-Hyun Cho, Jang-Eun Lee, Kwan Soo Hong, and Young-Shick Hong. Assessment of green coffee bean metabolites dependent on coffee quality using a 1h nmr-based metabolomics approach. *Food Research International*, 67:175–182, 2015.

[KPB18] A Kamilaris and FX Prenafeta-Boldú. A review of the use of convolutional neural networks in agriculture. *The Journal of Agricultural Science*, 156(3):312–322, 2018.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[LCWR15] Sue Han Lee, Chee Seng Chan, Paul Wilkin, and Paolo Remagnino. Deepplant: Plant identification with convolutional neural networks. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 452–456. IEEE, 2015.

[RCC15] Angie K Reyes, Juan C Caicedo, and Jorge E Camargo. Fine-tuning deep convolutional networks for plant recognition. In *CLEF (Working Notes)*, 2015.

[SDBR14] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[TAG13] Birhanu Turi, Getachew Abebe, and Girma Goro. Classification of ethiopian coffee beans using imaging techniques. *East African Journal of Sciences*, 7(1):1–10, 2013.

[YN10] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 270–279. ACM, 2010.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Exhibition and Evaluation of Two Schemes for Determining Hypersurface Curvature in Volumetric Data

Jacob D. Hauenstein
The University of Alabama in Huntsville
301 Sparkman Drive
Huntsville, AL 35899
hauensj1@uah.edu

Timothy S. Newman
The University of Alabama in Huntsville
301 Sparkman Drive
Huntsville, AL 35899
tnewman@cs.uah.edu

ABSTRACT

Advancements in methodologies for determining 3-dimensional manifold (hypersurface) curvature in volumetric data are presented. Such determinations are requisite in certain shape-based visualization and analysis tasks. The methods explored here are convolution-based approaches. In addition to motivating and describing these methods, an evaluation of their (1) accuracy and (2) computational performance is also presented. That evaluation includes benchmarking on both noise-free and noisy volumetric data.

Keywords

Curvature, Volumetric Data, 3D Manifold, Hypersurface, Imaging

1 INTRODUCTION

Curvature and related quantities have been found to be useful attributes for certain computer-based tasks utilizing range images (henceforth images), point clouds, or volumetric data (henceforth volumes or volume data). Application areas include computer graphics and visualization, healthcare, seismology, computational archaeology [Du18], etc. For example, curvature has been used in segmentation, object recognition, geometric modeling, and analysis of images and volumes [Bib16, Bes86, Bel12, Bag16, Lef18, Sou16], to perform reconstruction in images [Lef17], for biometrics [Sya17], for computer vision-based quality control in manufacturing [Kot18], etc. Other examples include emphasizing features in renderings of meshes [AR18] and images [Hau18], mesh parameterization [Vin17], highlighting shapes in volume renderings [Kin03], and visualization of medical data [Pre16]. In summary, curvature has been quite useful in computer graphics, visualization, and computer vision.

In fact, there is a rich literature that focuses on the use of curvature of 1-dimensional (1D) curves or 2-dimensional (2D) surfaces, including all of the works in the prior paragraph and [Lan07]. The works using curvature of conventional surfaces in 3-space are said, more formally, to operate on 2D manifolds (commonly simply called *surfaces*). In such domains, the maximum and minimum curvatures (known as the *principal curvatures* and denoted κ_1 and κ_2, respectively) can be valuable since they describe surface shape properties. These surfaces, despite existing in a 3D space, are 2D manifolds and thus have two principal curvatures.

Curvature of 3D manifolds (here called *hypersurfaces*, following the terminology used previously by Monga and Benayoun [Mon92]) also can be a useful descriptor for certain tasks, such as tasks operating on volumetric regions of data rather than on surface structures. For such tasks, the three principal curvatures, κ_1, κ_2, and κ_3 (ordered such that $\kappa_1 > \kappa_2 > \kappa_3$), can be of value. Some exploration of such curvatures in volume data has previously been described [Mon92, Ham94]. Recently, a number of works have reported use of these 3D manifold curvatures in medical assessment / segmentation, seismic data visualization, and surface reconstruction (e.g., [Ald14, Suz18, Pap07]). A previous work has also presented a framework capable of detecting crease surfaces in d-dimensional hypersurfaces [Yos12].

Another use for hypersurface curvatures is classification of volume data points based on the relative, relative absolute, and average values of these three curvature values [Hir01]. A table of all possible curvature classifications at each point has previously been presented (reproduced in part in Fig. 1), and has been used previously for visualization of volume shape [Hir01] and categorization of lung tumors [Hir18]. An example application of these curvature classifications and ensuing visualization (using the scheme from [Hir01]) on a CT scan of a phantom (CT scan obtained from http://www.santec.lu/project/optimage/samples)

No.	Simple	Absolute	Average
1	$\kappa_1>\kappa_2>\kappa_3>0$	$\lvert\kappa_1\rvert>\lvert\kappa_2\rvert>\lvert\kappa_3\rvert>0$	$\kappa_1+\kappa_2+\kappa_3>0$
2	$\kappa_1>\kappa_2>\kappa_3=0$	$\lvert\kappa_1\rvert>\lvert\kappa_2\rvert>\lvert\kappa_3\rvert=0$	$\kappa_1+\kappa_2+\kappa_3>0$
3	$\kappa_1>\kappa_2=\kappa_3>0$	$\lvert\kappa_1\rvert>\lvert\kappa_2\rvert=\lvert\kappa_3\rvert>0$	$\kappa_1+\kappa_2+\kappa_3>0$
4	$\kappa_1>\kappa_2=\kappa_3=0$	$\lvert\kappa_1\rvert>\lvert\kappa_2\rvert=\lvert\kappa_3\rvert=0$	$\kappa_1+\kappa_2+\kappa_3>0$
5	$\kappa_1=\kappa_2>\kappa_3>0$	$\lvert\kappa_1\rvert=\lvert\kappa_2\rvert>\lvert\kappa_3\rvert>0$	$\kappa_1+\kappa_2+\kappa_3>0$
6	$\kappa_1=\kappa_2>\kappa_3=0$	$\lvert\kappa_1\rvert=\lvert\kappa_2\rvert>\lvert\kappa_3\rvert=0$	$\kappa_1+\kappa_2+\kappa_3>0$
7	$\kappa_1=\kappa_2=\kappa_3>0$	$\lvert\kappa_1\rvert=\lvert\kappa_2\rvert=\lvert\kappa_3\rvert>0$	$\kappa_1+\kappa_2+\kappa_3>0$
8	$\kappa_1=\kappa_2=\kappa_3=0$	$\lvert\kappa_1\rvert=\lvert\kappa_2\rvert=\lvert\kappa_3\rvert=0$	$\kappa_1+\kappa_2+\kappa_3=0$
9	$\kappa_1>\kappa_2>0>\kappa_3$	$\lvert\kappa_1\rvert>\lvert\kappa_2\rvert>\lvert\kappa_3\rvert>0$	$\kappa_1+\kappa_2+\kappa_3>0$
⋮	⋮	⋮	⋮
39	$0>\kappa_1>\kappa_2>\kappa_3$	$\lvert\kappa_3\rvert>\lvert\kappa_2\rvert>\lvert\kappa_1\rvert>0$	$\kappa_1+\kappa_2+\kappa_3<0$
40	$0>\kappa_1>\kappa_2=\kappa_3$	$\lvert\kappa_2\rvert=\lvert\kappa_3\rvert>\lvert\kappa_1\rvert>0$	$\kappa_1+\kappa_2+\kappa_3<0$
41	$0>\kappa_1=\kappa_2>\kappa_3$	$\lvert\kappa_3\rvert>\lvert\kappa_2\rvert=\lvert\kappa_1\rvert>0$	$\kappa_1+\kappa_2+\kappa_3<0$
42	$0>\kappa_1=\kappa_2=\kappa_3$	$\lvert\kappa_1\rvert=\lvert\kappa_2\rvert=\lvert\kappa_1\rvert>0$	$\kappa_1+\kappa_2+\kappa_3<0$

Figure 1: A subset of the 42 possible classes of κ_1, κ_2, and κ_3 (adapted from [Hir01]).

is shown in Fig. 2. In the figure, a grayscale encoding of the classes is used, with class 1 mapped to black and with brightness increasing linearly to class 42, which is mapped to white.

However, to use hypersurface curvatures on sensed data, there needs to be some way to determine them. Here, we describe two methods for their determination. Since no comparative evaluation of hypersurface curvature determination methods has previously been performed, here we also present an evaluation of these two methods. Our study considers both the accuracy and run times of the methods, thereby providing insight into inaccuracies from and relative limitations in 3D manifold curvature determinations.

This paper is organized as follows. Section 2 provides background information on hypersurface curvature and some related surface curvature details. Section 3 describes the two hypersurface curvature determination methods we propose for volume data. Section 4 details our experimental procedures and results. Section 5 presents visualizations of curvature values in sensed data using one of the methods described. Section 6 concludes the work.

2 BACKGROUND AND PREVIOUS WORK

Here, we first provide an overview of the mathematics for determining curvature of hypersurfaces within volumetric data. Readers looking for a more detailed presentation may wish to see [Mon92] or [Ham94]. Then, we describe two methods for determining conventional surface curvature in volumetric datasets. (The two hypersurface curvature determination methods we describe later in this paper are inspired by these methods.)

The notation we use is as follows. (u,v,w) denotes a grid (or sample) point within the volume, where $0\leq u<N_u$, $0\leq v<N_v$, $0\leq w<N_w$ for a volume of size $N_u\times N_v\times N_w$. The value at point (u,v,w) is denoted $f(u,v,w)$; f represents the underlying function that generates the volume. Consequently, f_u represents the partial derivative of f in the u direction.

2.1 Hypersurface Curvature Mathematics

The hypersurface's three principal curvatures (i.e., of f) are the eigenvalues of the matrix:

$$\frac{1}{l}\begin{bmatrix} f_{uu} & f_{uv} & f_{uw} \\ f_{uv} & f_{vv} & f_{vw} \\ f_{uw} & f_{vw} & f_{ww} \end{bmatrix} \begin{bmatrix} 1+f_u^2 & f_u f_v & f_u f_w \\ f_u f_v & 1+f_v^2 & f_v f_w \\ f_u f_w & f_v f_w & 1+f_w^2 \end{bmatrix}^{-1}, \tag{1}$$

where

$$l=\sqrt{1+f_u^2+f_v^2+f_w^2}. \tag{2}$$

Thus, computation of κ_1, κ_2, and κ_3 requires knowledge of the first and second derivatives of f. For many curvature-based tasks on volumetric datasets, the continuous form of f is unknown because the data is sensed (i.e., acquired via a sensor). In such cases, the first and second derivatives must be estimated in order to determine κ_1, κ_2, and κ_3 based on evaluation of Eqn. 1. The two hypersurface curvature determination methods discussed later in this work both operate by estimating the necessary derivatives and using them in Eqn. 1.

2.2 Related Comparisons of Curvature Determination Methods

While we are not aware of any studies comparatively evaluating hypersurface curvature determination methods, some studies exist for related domains, for example of methods to determine surface curvature in range data [Bes86, Hau18]. In range data, many curvature determination methods exist, but those studies [Bes86, Hau18] found that no one method is uniformly best across all types of input. Method accuracy was found to vary depending on data type (e.g., noise-free synthetic or noisy sensed data) and surface shape. A previous study of surface curvature determination methods in volume data [Hau14] similarly concluded that no one method is best for all input data types.

2.3 Related Methods for Surface Curvature in Volumes

The two methods for determining hypersurface curvature proposed in this paper, described later, are volumetric analogues of two methods for determining surface curvature in volume data. Here, we describe those existing methods. Those methods determine surface curvature by (1) estimating derivatives at every point

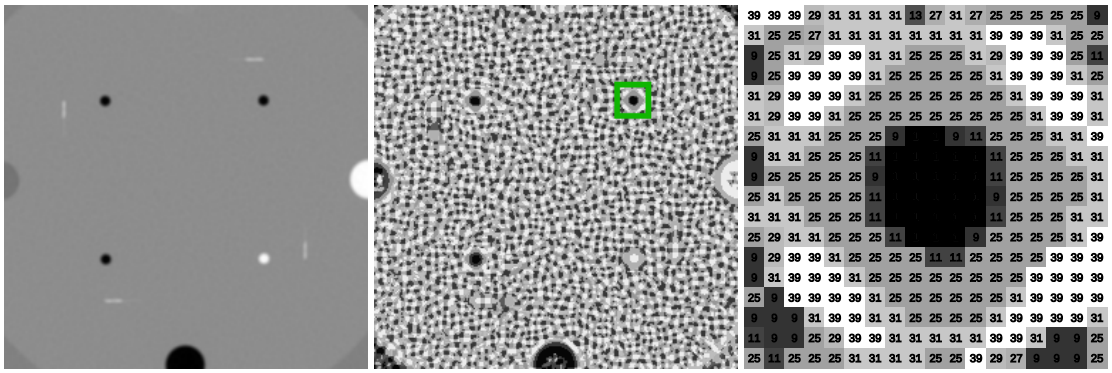


Figure 2: A slice from a CT scan of a phantom (left), the resulting data point-by-data point classifications of hypersurface curvatures according to Fig. 1 (middle), and zoomed-in detail (right). The classes are visualized (in middle and right) using a grayscale mapping.

in the volume and then (2) computing the curvature at every point in the volume using the estimated derivatives. We focus on these two existing methods because one of them was previously found to perform well on noise-free data, and one of them was previously found to perform well on noise-added and real data [Hau14], thus the volumetric analogues presented are based on two methods that, together, perform well across many types of input data.

2.3.1 Taylor Expansion

The first of those surface curvature methods was described by Kindlmann et al. [Kin03]. It uses separable convolution filters developed according to the framework of Möller et al. [Möl98]. Such filters are based on the Taylor Expansion and allow for a given accuracy and continuity.

The Kindlmann et al. method uses these first and second derivative filters at every data point in the volume to estimate the derivatives of f required for computation of surface curvature. Then, these estimates of f's derivatives are used to determine gradients, Hessians, and, finally, curvatures. Previous studies [Kin03, Hau14] have found a Taylor Expansion-based strategy to be very fast and accurate for surface curvature determination on synthetic, noise-free data.

2.3.2 Orthogonal Polynomials

The second of those surface curvature methods for determining surface curvature in volumes was described by Hauenstein and Newman [Hau14]. It was motivated by a method for determining surface curvature in range images described by Besl and Jain [Bes86] based on a locally fit surface model [Fly89]. To estimate derivatives, it uses separable convolution filters sampled from orthogonal polynomials. Such filters implicitly perform a least squares fitting [Fly89].

Like the previously discussed Taylor Expansion-based method for determining surface curvature, the Hauenstein and Newman method uses convolution to estimate first and second derivatives and then uses these derivative estimates to determine curvature. A previous study [Hau14] found this orthogonal polynomials-based method to be relatively accurate and fast for surface curvature determination on noise-added synthetic volumes as well as sensed volumes.

3 HYPERSURFACE CURVATURE DETERMINATION METHODS

Here, we describe in detail our proposed methods for determining hypersurface curvature within volume data.

3.1 Taylor Expansion Filters (TEF)

The first hypersurface curvature method we describe, denoted **TEF**, uses convolution filters derived from the Taylor Expansion. It is a hypersurface analogue of the Kindlmann et al. [Kin03] surface curvature determination approach. The method determines hypersurface curvatures by first performing a series of convolutions along each axis. These convolutions provide estimates for first and second directional derivatives. Once these derivatives are known, the three principal curvatures are computed as the eigenvalues of Eq. 1.

The method's filters are devised according to the Möller et al. framework [Möl98]. The filters have accuracy and continuity parameters, and the chosen accuracy and continuity parameters also impact the size of the resulting convolution filters. Here, we use filters with C^3 continuity and fourth order accuracy, and they thus allow for exact reconstruction of functions of degree 3 or lower. The first derivative filter is:

$$\{-\frac{1.0}{12.0}, \frac{2.0}{3.0}, 0.0, -\frac{2.0}{3.0}, \frac{1.0}{12.0}\}. \quad (3)$$

The second derivative filter is:

$$\{-\frac{1.0}{24.0}, \frac{1.0}{6.0}, \frac{17.0}{24.0}, -\frac{5.0}{3.0}, \frac{17.0}{24.0}, \frac{1.0}{6.0}, -\frac{1.0}{24.0}\}. \quad (4)$$

Convolving the appropriate filters in the appropriate directions allows these filters to estimate all necessary

first, second, and mixed partial derivatives. Specifically, f_i is found by convolving the first derivative filter in the i direction. Other first derivatives are found analogously. f_{ij} is found by convolving the first derivative filter in both the i and j directions. Other mixed partials are found analogously. f_{ii} is found by convolving the second derivative filter in the i direction. Other second derivatives are found analogously.

These filters were previously found by Kindlmann et al. [Kin03] to perform well in determination of conventional surface curvature in volumes. Thus, we were motivated to extend them to hypersurface curvature.

3.2 Orthogonal Polynomials Filters (OPF)

The second hypersurface curvature method we describe, denoted **OPF**, also uses a series of 1D convolutions to determine directional derivatives. And, like **TEF**, **OPF** uses these estimated derivatives to compute the three principal curvatures as the eigenvalues of Eq. 1. Unlike **TEF**, the convolution filters for **OPF** are derived from orthogonal polynomials. Such filters implicitly perform a least squares fitting [Bes86, Fly89], and the derivatives estimated via this method are thus identical to those of a local surface that well-fits that neighborhood (i.e., via linear regression).

In **OPF**, derivative estimation filters of odd size N are used. These filters are generated by sampling orthogonal polynomials, b_0, b_1, b_2, at N locations (more details about these polynomials are in [Bes86]):

$$b_0(\theta) = \frac{1}{N}, \tag{5}$$

$$b_1(\theta) = \frac{3}{M(M+1)(2M+1)}\theta, \tag{6}$$

$$b_2(\theta) = \frac{1}{P(M)}(\theta^2 - \frac{M(M+1)}{3}), \tag{7}$$

where $M = \frac{N-1}{2}$, $P(M)$ is given by:

$$P(M) = \frac{8}{45}M^5 + \frac{4}{9}M^4 + \frac{2}{9}M^3 - \frac{1}{9}M^2 - \frac{1}{15}M, \tag{8}$$

and θ denotes the locations at which each polynomial is sampled, with $\theta \in \{-\frac{N-1}{2}, ..., -1, 0, 1, ..., \frac{N-1}{2}\}$. Applying the b_0 filter performs smoothing. Applying the b_1 filter generates estimates of the first derivative. Applying the b_2 filter generates estimates of the second derivative.

Using these kernels, estimated partial and mixed partial derivatives of f are found. Specifically, to find f_i, $i \in \{u, v, w\}$, we (1) convolve in the i direction with the discrete filter resulting from sampling b_1 and (2) convolve in each of the other two directions with the discrete filter resulting from sampling b_0. Moreover, to find f_{ij}, $i, j \in \{u, v, w\}$ and $i \neq j$, we (1) convolve in the i and j directions with the discrete filter resulting from sampling b_1 and (2) convolve in the other direction with the discrete filter resulting from sampling b_0. Lastly, to find f_{ii}, $i \in \{u, v, w\}$, we (1) convolve in the i direction with the b_2 filter sampling and (2) convolve in the other directions with the b_0 filter sampling. Once all necessary derivatives are estimated through this process, the hypersurface curvatures are calculated as the eigenvalues of Eq. 1.

We have used a filter size, N, of 7 for our applications here, since a prior work [Hau14] found $N = 7$ to be appropriate for surface curvature within volumes.

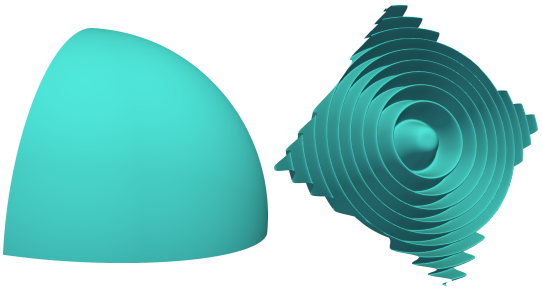


Figure 3: An isosurface at 10000 from the Spheres dataset (left), and an isosurface at 0 from the ML dataset (right).

4 EXPERIMENTS AND RESULTS

In this section, we describe the experiments in and results of our comparative evaluation of the **TEF** and **OPF** hypersurface curvature methods. The experiments involve both accuracy and run time tests. Synthetic volume datasets are used in these tests. We next describe those datasets.

4.1 Synthetic Datasets

Noise-free and noise-added versions of two synthetic volumes were used. Each was sampled from a continuous volume. Since the continuous form of these volumes are known, the three principal curvatures of the continuous forms can be computed exactly. Our experiments evaluate accuracy by comparing these known curvatures to the curvatures determined by the methods.

The first dataset, called Spheres, is generated using the function:

$$f(u, v, w) = (u^2 + v^2 + w^2). \tag{9}$$

The formula was sampled on a $256 \times 256 \times 256$ grid with each axis in the range $[0, 255]$, resulting in a volume containing values ranging from 0 to 195075. Our experiments consider this volume with and without added Gaussian noise ($\mu = 0$, $\sigma = 0.001$). A rendering of a level surface of this dataset is shown in Fig. 3 (left).

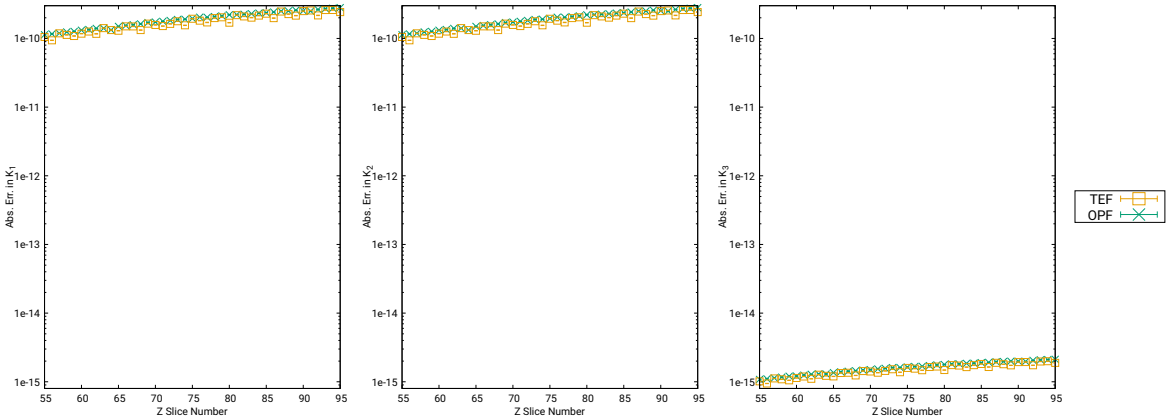


Figure 4: Per-slice average error in κ_1, κ_2, and κ_3 for noise-free Spheres.

	κ_1 Avg. Abs. Err.	κ_1 Std. Err.	κ_2 Avg. Abs. Err.	κ_2 Std. Err.	κ_3 Avg. Abs. Err.	κ_3 Std. Err.
TEF	$3.74 \cdot 10^{-10}$	$2.21 \cdot 10^{-13}$	$3.74 \cdot 10^{-10}$	$2.21 \cdot 10^{-13}$	$2.01 \cdot 10^{-15}$	$8.46 \cdot 10^{-19}$
OPF	$4.09 \cdot 10^{-10}$	$2.32 \cdot 10^{-13}$	$4.09 \cdot 10^{-10}$	$2.32 \cdot 10^{-13}$	$2.18 \cdot 10^{-15}$	$8.73 \cdot 10^{-19}$

Figure 5: Global absolute average errors and standard errors for noise-free Spheres.

The second dataset, called ML, is generated from a function originally defined by Marschner and Lobb [Mar94]. This function is of particular interest because it is known to be band-limited but demanding on reconstruction tasks [Mar94]. The ML function has the form:

$$f(u,v,w) = \frac{(1-\sin(\pi w/2)+\beta(1+\rho(\sqrt{u^2+v^2})))}{2(1+\beta)} - 0.5, \quad (10)$$

where ρ is the function given by:

$$\rho(r) = \cos(2\pi f_M \cos(\frac{\pi r}{2})). \quad (11)$$

In the experiments we report here, we used $\beta = 0.25$ with $f_M = 6$, which Marschner and Lobb chose because it placed a significant amount of the function's energy near the Nyquist frequency when sampled on a $40 \times 40 \times 40$ grid in the range $[-1.0, 1.0]$ [Mar94]. For our experiments, we sampled the function on a $256 \times 256 \times 256$ grid in the range $[-1.0, 1.0]$, resulting in a volume containing values ranging from -0.5 to 0.5. Our experiments consider this volume with and without added Gaussian noise ($\mu = 0$, $\sigma = 0.001$). A rendering of a level surface from this volume is shown in Fig. 3 (right).

4.2 Accuracy Results

Accuracy experiment results (for the two methods) on the noise-free Spheres dataset are shown in Fig. 4. The figure shows the average absolute errors along a subset of Z slices. We note that since the methods exhibit very large error at edges of the dataset due to the convolution filters extending beyond the edges of the dataset, these plots exclude points within 10 units from the edges; the plots are representative of the typical errors when the filters have support. Fig. 5 presents a table of the average global errors (again only in locations with support) and standard errors.

For this simple noise-free dataset, both **TEF** and **OPF** produce results with very low error, with similar error levels for κ_1 and κ_2 and smaller error levels for κ_3. This outcome is not unexpected since κ_1 and κ_2 are equal for this dataset and κ_3 is much smaller. For all three curvatures, the **TEF** method exhibits slightly lower error.

Accuracy results for the noise-added Spheres dataset are shown in Fig. 6 and Fig. 7. In this case, Gaussian noise was added to the dataset prior to determining curvatures. In this noise-added case, despite the relatively small amount of noise, the **TEF** method exhibits more error than the **OPF** method. (A similar difference in error was noted for use of a Taylor Expansion-based strategy to determine conventional surface curvatures in volume data [Hau14].)

The average absolute errors for the noise-free ML dataset are shown in Fig. 8 and Fig. 9. Unlike Spheres, which can be exactly locally fit with a polynomial, this sinusoidal dataset cannot be exactly locally fit with a polynomial. Consequently, it is not surprising that **TEF** yielded more accurate curvatures since it employs higher degree polynomials (compared to **OPF**).

The average absolute errors for the noise-added ML are shown in Fig. 10 and Fig. 11. While in the noise-free case **TEF** exhibited much lower error compared to **OPF**, in this noise-added case **OPF** has much lower error. This is most likely due to the smoothing that implicitly occurs as a part of convolution with orthogonal polynomials.

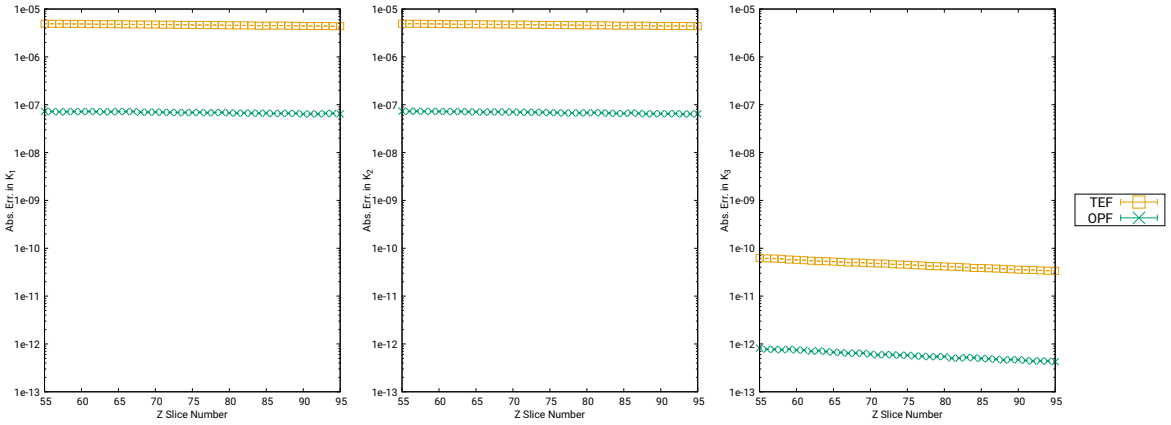


Figure 6: Per-slice average error in κ_1, κ_2, and κ_3 for noise-added Spheres.

	κ_1 Avg. Abs. Err.	κ_1 Std. Err.	κ_2 Avg. Abs. Err.	κ_2 Std. Err.	κ_3 Avg. Abs. Err.	κ_3 Std. Err.
TEF	$4.04 \cdot 10^{-06}$	$1.04 \cdot 10^{-09}$	$4.04 \cdot 10^{-06}$	$1.04 \cdot 10^{-09}$	$3.88 \cdot 10^{-11}$	$7.23 \cdot 10^{-14}$
OPF	$5.97 \cdot 10^{-08}$	$1.49 \cdot 10^{-11}$	$5.97 \cdot 10^{-08}$	$1.49 \cdot 10^{-11}$	$5.00 \cdot 10^{-13}$	$8.76 \cdot 10^{-16}$

Figure 7: Global absolute average errors and standard errors for noise-added Spheres.

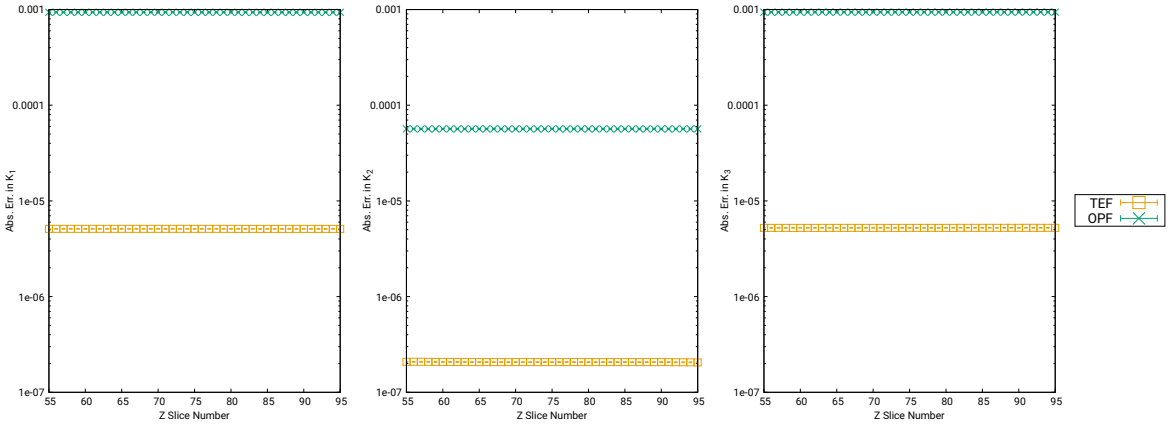


Figure 8: Per-slice average error in κ_1, κ_2, and κ_3 for noise-free ML.

	κ_1 Avg. Abs. Err.	κ_1 Std. Err.	κ_2 Avg. Abs. Err.	κ_2 Std. Err.	κ_3 Avg. Abs. Err.	κ_3 Std. Err.
TEF	$5.09 \cdot 10^{-06}$	$2.31 \cdot 10^{-09}$	$2.09 \cdot 10^{-07}$	$1.23 \cdot 10^{-10}$	$5.24 \cdot 10^{-06}$	$2.39 \cdot 10^{-09}$
OPF	$9.34 \cdot 10^{-04}$	$3.74 \cdot 10^{-07}$	$5.66 \cdot 10^{-05}$	$2.08 \cdot 10^{-08}$	$9.42 \cdot 10^{-04}$	$3.75 \cdot 10^{-07}$

Figure 9: Global absolute average errors and standard errors for noise-free ML.

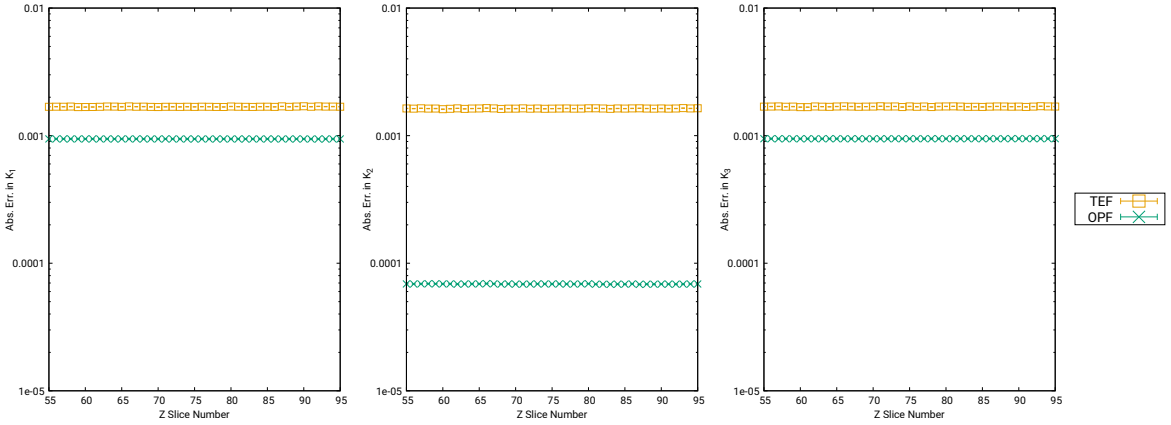


Figure 10: Per-slice average error in κ_1, κ_2, and κ_3 for noise-added ML.

	κ_1 Avg. Abs. Err.	κ_1 Std. Err.	κ_2 Avg. Abs. Err.	κ_2 Std. Err.	κ_3 Avg. Abs. Err.	κ_3 Std. Err.
TEF	$1.68 \cdot 10^{-03}$	$3.50 \cdot 10^{-07}$	$1.63 \cdot 10^{-03}$	$3.40 \cdot 10^{-07}$	$1.69 \cdot 10^{-03}$	$3.50 \cdot 10^{-07}$
OPF	$9.40 \cdot 10^{-04}$	$3.73 \cdot 10^{-07}$	$6.87 \cdot 10^{-05}$	$1.89 \cdot 10^{-08}$	$9.48 \cdot 10^{-04}$	$3.74 \cdot 10^{-07}$

Figure 11: Global absolute average errors and standard errors for noise-added ML.

	Noise-Free Spheres	Noise-Added Spheres	Noise-Free ML	Noise-Added ML
TEF	58.30	64.32	57.60	64.98
OPF	103.30	106.84	103.10	106.72

Figure 12: Average run times (seconds).

Figure 13: An isosurface of a CT scan of the Stanford Bunny colored using a grayscale mapping of hypersurface curvature classes.

4.3 Run Time Results

Next, we report on computational performance of the methods on the four $256 \times 256 \times 256$ datasets. Both methods were implemented in single-threaded C++ with the Armadillo library used for linear algebra operations. Testing was done on a machine running GNU/Linux equipped with 16GB of DDR3 RAM and an Intel i5-2310 processor. All code was compiled with g++. Run time (computational performance) results for each method on each dataset are reported in Fig. 12. These times represent the trimmed means of 10 runs with the fastest and slowest runs excluded.

TEF is substantially faster than **OPF**. Since both methods use the same linear algebra library, the time variation between the two is due only to the convolution step. While **OPF** convolves along all three axes at each point, even when measuring a derivative in only one direction, **TEF** only convolves along axes on which derivatives are being estimated. As a result, fewer convolutions are performed in the **TEF** method, and it exhibits much faster run time.

5 VISUAL RESULTS

The superior resilience of **OPF** in the presence of noise motivates the selection of it when utilizing hypersurface curvatures on real data, and for that reason we have used its curvatures to produce the curvature classifications shown previously in Fig. 2, and we have additionally used it in related curvature classification visualization tasks utilizing sensed data. We describe those visualization tasks and their results next.

First, we consider a CT scan of the Stanford bunny (obtained from http://graphics.stanford.edu/data/voldata/voldata.html). In Fig. 13, we present a rendering of an isosurface of the bunny. This isosurface is colored using a grayscale mapping of hypersurface curvature classifications. The rendering thus presents a 2D

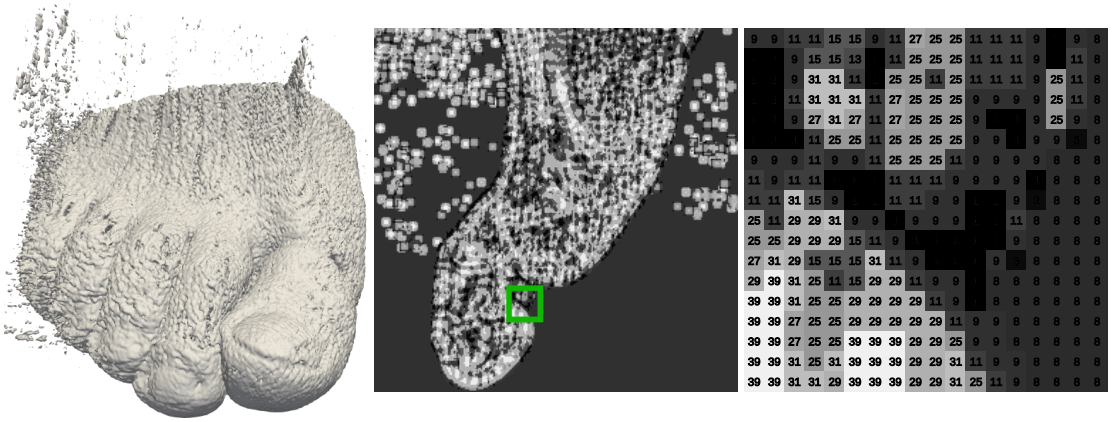


Figure 14: An isosurface rendering of a CT scan of a human foot (left), the resulting data point-by-data point classifications of hypersurface curvatures according to Fig. 1 (middle) for one slice, and zoomed-in detail (right) for that slice. The classes are visualized (in middle and right) using a grayscale mapping.

manifold (the isosurface) colored using the curvature values of the 3D manifold present at each isosurface point. The grayscale mapping used is identical to the one used in the middle and right of Fig. 2. As this is sensed data, some noise is clearly present in the classification results (e.g., a notable stair-stepping can be seen on the face and ears of the bunny), however, many features are also clearly indicated from changes in hypersurface curvature classification, including the bunny's whiskers, eye socket, and toes.

Second, we consider a CT scan of a human foot (obtained from http://volvis.org). An isosurface rendering of the foot is shown in the left part of Fig. 14. In the middle of that figure, grayscale mapped hypersurface curvature classifications of z-slice 127 (located on the index toe next to the big toe) of the dataset are shown (again using the same mapping as in Fig. 2). The green box shown in the middle of the figure represents the location of the zoomed-in region shown in the right of the figure. This CT scan exhibits much noise, as can be seen in both the isosurface rendering and the hypersurface curvature classifications. Consequently, it is difficult to visually locate fine features within the slice.

6 CONCLUSION

In this paper we have exhibited two methods for determining hypersurface curvature within volume datasets. Both of these methods use convolution to estimate derivatives and then these estimated derivatives are used to determine the three hypersurface curvatures. One method, **TEF**, uses convolution with filters based on the Taylor Expansion. The other, **OPF**, uses convolution with filters sampled from orthogonal polynomials. We have additionally comparatively evaluated both these methods in terms of (1) accuracy and (2) run time.

In summary, in our tests we found that **TEF** exhibits relatively low error levels when no noise is present. However, when (even relatively low level) noise is present, **OPF** generally exhibits much lower error levels than **TEF**. However, **TEF** exhibits much faster run times than **OPF** in both noise-free and noise-added cases. Thus, if time is critical or no noise is present in the dataset, **TEF** is the preferable choice. If noise is present and accuracy is critical, **OPF** is a better choice.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for contributing their input and insight. These contributions helped to greatly improve the paper.

7 REFERENCES

[Ald14] G. Aldrich, A. Gimenez, M. Oskin, R. Strelitz, J. Woodring, L. H. Kellogg, and B. Hamann. Curvature-based crease surfaces for wave visualization. *Vision, Modeling & Vis.*, pp. 39–46, 2014.

[AR18] R. Al-Rousan, M. S. Sunar, and H. Kolivand. Geometry-based shading for shape depiction enhancement. *Multimedia Tools and Appl.*, 77(5):5737–5766, Mar 2018.

[Bag16] P. Bagchi, D. Bhattacharjee, and M. Nasipuri. A robust analysis, detection and recognition of facial features in 2.5d images. *Multimedia Tools and Appl.*, 75(18):11059–11096, Sep 2016.

[Bel12] A. Belyaev. Focal surfaces, skeletons, and ridges for shape analysis and processing. *Proceedings of the 2012 Joint International Conference on Human-Centered Computer Environments*, HCCE '12, pp. 58–58, 2012.

[Bes86] P. J. Besl and R. C. Jain. Invariant surface characteristics for 3d object recognition in range images. *Comput. Vision, Graphics, Image Processing*, 33(1):33 – 80, 1986.

[Bib16] P. Bibiloni, M. González-Hidalgo, and S. Massanet. A survey on curvilinear ob-

ject segmentation in multiple applications. *Pattern Recognition*, 60:949–970, Dec 2016.

[Du18] G. Du, C. Yin, M. Zhou, Z. Wu, and F. Duan. Part-in-whole matching of rigid 3D shapes using geodesic disk spectrum. *Multimedia Tools and Appl.*, 77(15):18881–18901, Aug 2018.

[Fly89] P. Flynn and A. Jain. On reliable curvature estimation. *Proc., IEEE Comput. Vision and Pat. Recog.*, pp. 110 – 116, 1989.

[Ham94] B. Hamann. Curvature approximation of 3D manifolds in 4D space. *Comput. Aided Geometric Design*, 11(6):621 – 632, 1994.

[Hau14] J. D. Hauenstein and T. S. Newman. On reliable estimation of curvatures of implicit surfaces. *Proc., 2nd Int'l Conf. 3D Vision (3DV)*, pp. 697 – 704, Dec 2014.

[Hau18] J. D. Hauenstein and T. S. Newman. Curvature determination in range images: new methods and comparison study. *Multimedia Tools and Appl.*, Advance online publication, Aug 2018.

[Hir01] Y. Hirano, A. Shimizu, J.-i. Hasegawa, and J.-i. Toriwaki. A tracking algorithm for extracting ridge lines in three-dimensional gray images using curvature of four-dimensional hypersurface. *Systems and Comput. in Japan*, 32(12):25–37, 2001.

[Hir18] Y. Hirano. Categorization of lung tumors into benign/malignant, solid/GGO, and typical benign/others. K. Suzuki and Y. Chen, editors, *Artificial Intelligence in Decision Support Systems for Diagnosis in Medical Imaging*, pp. 193–208. 2018.

[Kin03] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Möller. Curvature-based transfer functions for direct volume rendering: methods and applications. *Proc., Vis. '03*, pp. 513 – 520, 2003.

[Kot18] K. Kottari, K. Delibasis, and V. Plagianakos. Real time vision-based measurements for quality control of industrial rods on a moving conveyor. *Multimedia Tools and Appl.*, 77(8):9307 – 9324, Apr 2018.

[Lan07] T. Langer, A. Belyaev, and H.-P. Seidel. Exact and interpolatory quadratures for curvature tensor estimation. *Computer Aided Geometric Design*, 24(8):443 – 463, 2007.

[Lef17] D. Lefloch, M. Kluge, H. Sarbolandi, T. Weyrich, and A. Kolb. Comprehensive use of curvature for robust and accurate online surface reconstruction. *IEEE T-Pat. Anal. Machine Int.*, 39(12):2349–2365, Dec 2017.

[Lef18] L. Lefkovits and S. Lefkovits. Two-phase MRI brain tumor segmentation using random forests and level set methods. *Short Papers Proc., Int'l Conf. in Central Europe on Comput. Graphics, Vis. and Comput. Vision (WSCG2018)*, pp. 152–159, 2018.

[Mar94] S. Marschner and R. Lobb. An evaluation of reconstruction filters for volume rendering. *Proc., Vis. '94*, pp. 100 – 107, 1994.

[Möl98] T. Möller, K. Mueller, Y. Kurzion, R. Machiraju, and R. Yagel. Design of accurate and smooth filters for function and derivative reconstruction. *Proc., IEEE Symp. Volume Vis. 1998*, pp. 143 – 151, 1998.

[Mon92] O. Monga and S. Benayoun. Using partial derivatives of 3D images to extract typical surface features. Research Report RR-1599, INRIA, 1992.

[Pap07] L. Papaleo. An approach to surface reconstruction using uncertain data. *Int'l J. of Image and Graphics*, 07(01):177–194, 2007.

[Pre16] B. Preim, A. Baer, D. Cunningham, T. Isenberg, and T. Ropinski. A survey of perceptually motivated 3D visualization of medical image data. *Comput. Graphics Forum*, 35(3):501–525, 2016.

[Sou16] M. Soufi, H. Arimura, K. Nakamura, F. P. Lestari, F. Haryanto, T.-a. Hirose, Y. Umedu, Y. Shioyama, and F. Toyofuku. Feasibility of differential geometry-based features in detection of anatomical feature points on patient surfaces in range image-guided radiation therapy. *Int'l J. of Comput. Assisted Radiology and Surgery*, 11(11):1993–2006, 2016.

[Suz18] H. Suzuki, Y. Kawata, N. Niki, T. Sugiura, N. Tanabe, M. Kusumoto, K. Eguchi, and M. Kaneko. Automated assessment of aortic and main pulmonary arterial diameters using model-based blood vessel segmentation for predicting chronic thromboembolic pulmonary hypertension in low-dose ct lung screening. *Medical Imaging 2018: Comput.-Aided Diagnosis*, volume 10575, 2018.

[Sya17] M. A. Syarif, T. S. Ong, A. B. J. Teoh, and C. Tee. Enhanced maximum curvature descriptors for finger vein verification. *Multimedia Tools and Appl.*, 76(5):6859–6887, Mar 2017.

[Vin17] A. M. Vintescu, F. Dupont, and G. Lavoué. Least squares affine transitions for global parameterization. *Journal of WSCG*, 25(01):21–30, 2017.

[Yos12] S. Yoshizawa, A. Belyaev, and H. Yokota. Shape and image interrogation with curvature extremalities. *J. for Geometry and Graphics*, 16:81–95, Jan 2012.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Accounting for Object Weight in Interaction Design for Virtual Reality

Jesper Rask Lykke, August Birk Olsen, Philip Berman, J. Andreas Bærentzen, Jeppe Revall Frisvad
Department of Applied Mathematics and Computer Science, Technical University of Denmark
Richard Petersens Plads, DTU - Building 324
2800 Kongens Lyngby, Denmark
jerf@dtu.dk

ABSTRACT

Interaction design for virtual reality (VR) rarely takes the weight of an object – let alone its moment of inertia – into account. This clearly reduces user immersion and could lead to a break-in-presence. In this work, we propose methods for providing a higher fidelity in interactions with virtual objects. Specifically, we present different methods for picking up, handling, swinging, and throwing objects based on their weight, size, and affordances. We conduct user studies in order to gauge the differences in performance as well as sense of presence of the proposed techniques compared to conventional interaction techniques. While these methods all rely on the use of unmodified VR controllers, we also investigate the difference between using controllers to simulate a baseball bat and swinging a real baseball bat. Interestingly, we find that realism of the motions during interaction is not necessarily an important concern for all users. Our modified interaction techniques, however, have the ability to push user performance towards the slower motions that we observe when a real bat is used instead of a VR controller on its own.

1 INTRODUCTION

Regardless of the immersivity of a virtual reality experience, the body of the user of course never leaves the actual reality. This becomes a concern when we interact virtually with objects that appear to have very different physical properties from the controllers that we actually hold in our hands. In particular, virtual objects may have an expected mass and moment of inertia that is very different from the very light controllers. Nevertheless, we often employ a very direct *mapping* [Jer16] from the position and orientation of the controller to corresponding properties for virtual objects such as long sticks or heavy stones.

Studies have shown that a natural control mapping leads to more immersive interactions, which makes for a more enjoyable experience [SCP11]. The most physically realistic control mapping is known as *realistic tangible mapping* which makes use of a physical analogue to whichever object the user is interacting with [STS*11]. Shooting at digital targets with a gun-shaped controller, like in old arcade games, is one such example. While this control type may be the most realistic, it is not a viable method in applications where the user will interact with several different objects of varying shape, size, and functionality.

In lieu of a physical controller for every interactive object, a modified controller mapping might be used to simulate the physical properties and indicate the affordances of the objects. This is our point of focus.

To evaluate the influence of our techniques on the level of spatial presence, we use performance tests, and in some experiments also the Temple Presence Inventory (TPI) [LDW09]. The TPI method was developed and validated using psychological measurement procedures and has the purpose of quantifying the user's dimensions of presence. We thus use it for measuring spatial presence, but we also extend the questionnaire to provide a measure of how enjoyable the interactions felt to the test participants. Our performance tests measure how objects are handled. To assess the realism of a user's interaction with an object, we measure user performance with the controller attached to a baseball bat. We observe the qualitative change in user performance with and without a real bat, and we use this as a guideline for assessing the impact of our interaction techniques with respect to interaction realism.

To experiment with the handling and weight of objects, we set up a test scene containing a lightweight sword and a heavy hammer. To add to the atmosphere and interactivity, an axe, torches, and a shield were also included. Using the virtual weight of objects, we test unphysical downscaling of the controller velocity upon

transfer of this to the in-flight velocity of a thrown object. To assess the effect of this and of the size and appearance of objects, we also perform a test where users throw different balls into a barrel. Our performance tests are hidden within small games to sustain user interest and immersion during each test. In a baseball bat test, we use a crude baseball stadium with a pitching machine as the test environment.

2 RELATED WORK

Simeone et al. [SVG15] investigated the change in people's spatial presence when presented with virtual objects of different levels of mismatch to that of the physical world. They found that having virtual objects that correspond almost one-to-one with physical objects (e.g. a flashlight as a lightsaber) would greatly increase the levels of spatial presence. However, they also found that a mismatch in weight or fragility would be particularly undesirable (unless we want to simulate super-human strength). However, setting up a one-to-one mapping between virtual and physical objects is in many cases problematic.

A technique like haptic retargeting [AHB*16] can be employed to let one object represent multiple virtual objects, but the weight of the object will be unchanged. As the moment of inertia of an object is directly related to its perceived weight and length, a device offering dynamic weight-shifting capabilities can significantly enhance perceived changes in shape and weight of virtual objects [ZK17]. Haptic devices applying gravity-like forces to the hand grabbing a virtual object can perhaps increase the level of realism even further [MFK*07, CCM*17]. These devices are however significantly beyond what standard VR controllers can do. To explore our options when such supplementary devices are not available, we consider software solutions. Our assumption is that the interaction design can influence the user experience in a way so that the objects (to some degree) seem to have a different weight and shape even if the controllers are unchanged. Software-based object interaction will obviously not feel as realistic as substitutional reality or haptic solutions, but it may take the user some of the way in cases where other solutions are not available.

As explored by Dominjon et al. [DLB*05], it is certainly possible to influence a user's perception of the mass of a grabbed object. This can be done by modifying the visual motion of the object controlled by the user. Motion amplification leads to a feeling of a lower mass and, conversely, motion damping leads to a feeling of a higher mass. This means that an unmodified VR controller can potentially provide an impression of interacting with objects of different virtual weight. Animating a self-avatar to reflect the weight of virtual objects that the user interacts with is one way to influence the perceived object weight [JAO*14]. Our methods directly modify the user-object interactions without use of a self-avatar.

Figure 1: Our main test scene.

The work most closely related to ours is that of Rietzler et al. [RGGR18]. They present a VR interaction technique that modifies the motion of the virtual object based on its weight. A grabbed object then no longer directly follows the user's manipulation of the controller. This is similar to the "chasing" method that we propose in the following. On the other hand, our test cases, comparison to substitutional reality, and our other methods are quite different from their work.

3 TEST SCENE

An overview of our main test scene is shown in Figure 1. Every object, apart from the table and the viking, can be picked up. The torch flames and the viking are animated. As the scene has a fairly realistic appearance, the test participants should expect a realistic interaction design. The implemented affordances and functionality will be explained in the following.

4 INTERACTIONS

We first describe our use of handle points. These are points on an object that people would normally grab when using the object as intended (such as the hilt of a sword).

4.1 One-Handed Interactions

When picking up an object with no handle points, the point of collision between the end of the controller wand and the object becomes a fixed joint. While picked up, the transformations of the controller are applied to the object, so that the same relative distance between controller and object is kept as illustrated in Figure 2. For heavier objects with no handle points, we introduce an alternative type of handling called scooping. This is further explained in Section 6.2.

Picking up objects with a handle point is a bit more involved. If the controller is within a certain distance from the object's handle point when the trigger is pressed, the object is translated so the handle point is at the same position as the controller, and its rotation is

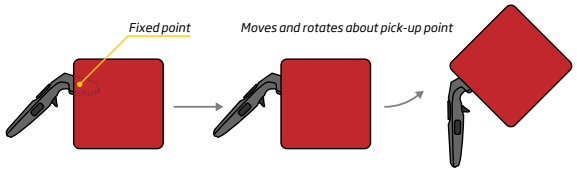


Figure 2: Object with no handle points.

set to that of the controller. When the object is in place, a fixed joint is created between object and controller.

If the controller is too far away from the handle point when picking up the object, a configurable joint is created between the controller and selected object. This type of joint gives the effect of lightly holding the object as if between the thumb and index finger, since it can be moved, but will always rotate its center of mass toward the ground.

When the trigger button is released, a release method checks if the selected object is held by the other controller. If this is the case, the releasing controller simply loses control of the object. If not, the joint between controller and object is removed and the velocity of the object is set to the velocity of the controller at the moment of release divided by the virtual mass of the object. This is unphysical, but it means that a user will not be able to throw heavy objects as far as lighter ones.

4.2 Two-Handed Interactions

If a user picks up a two-handed object outside the reach of a handle point, we create a configurable joint as for objects with one handle point. Alternatively, if a handle point is within reach, the controller grabs it. The object is then moved so that the handle point is at the position of the controller and a configurable joint is created. When the second controller grabs the object, the configurable joint is removed and the object is wielded with two hands.

A wielded, two-handed object is translated so that its first handle point is at the position of the controller that grabbed it first (Figure 3, left). The object is then rotated using a quaternion to have its forward vector pointing toward the second controller (Figure 3, middle). Since the local axis of our objects point upward by default, we apply a 90° rotation about the object's local x-axis. Finally, in order to be able to turn the object around its handle, the rotation of the last controller that grabbed the object is concatenated with the rotation of the object (Figure 3, right). The rotations are visualized in Figure 3.

Letting go of a two-handed object works in roughly the same way as with a one-handed object. When the trigger button is released on one of the controllers, the velocity of the controller is transferred to the object and the user loses control of it.

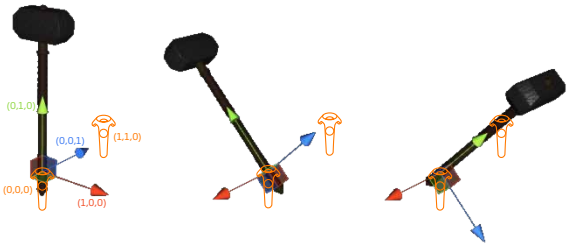


Figure 3: The rotations performed to make the two-handed object position itself between the controllers.

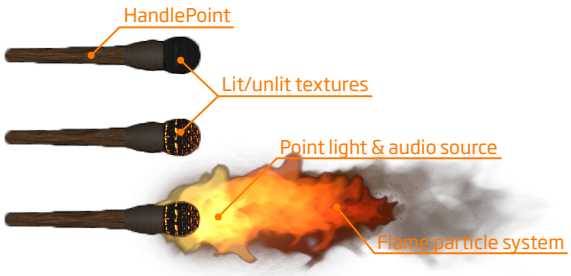


Figure 4: Torch components.

5 AFFORDANCES

5.1 Torch Light

Torches were implemented as both an interactive object and as a way of lighting the scene. The scene has a total of four torches: one on the floor and three hanging from holders on the walls. The torch on the floor is on fire from the beginning, giving the indication that all torches may be lit. Flames were added using noise-based particle systems. The torch components are illustrated in Figure 4.

The user is able to pick up the torch on the floor and take it with them as they walk around the room, using it to illuminate anything they wish to inspect. They can also use it to ignite the remaining torches that are hanging on the walls. Those torches are held in place by fixed joints, but if the user attempts to pick one of them up, the joint will be overridden by the one created by the user interaction.

5.2 Shield Orientation

The shield is meant to be carried on the side of the arm, which is why it has both a handle and an armstrap as shown in Figure 5. When it is picked up by the handle, the shield is rotated so the arm strap moves roughly to where the users arm would appear inside the scene.

The orientation of the shield is different depending on which hand is used to pick it up. A function checks which controller that picked up the object and rotates the shield accordingly. This is an example of an object that must be oriented differently for each hand.

5.3 Sword and Hammer

To compare one- and two-handed weapon interactions, a light sword and a heavy hammer were added to the

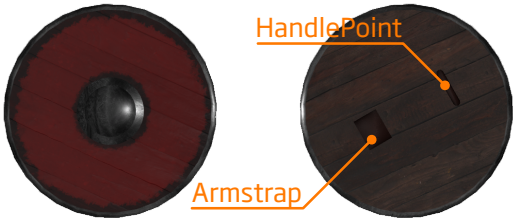


Figure 5: Front and back of the shield.

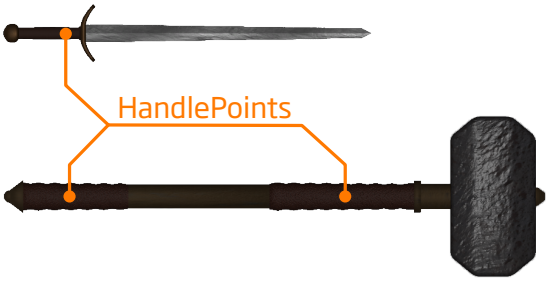


Figure 6: Handle points on sword and hammer.

scene. These two objects vary greatly both in size and material properties. The sword was modeled with a narrow hilt and thin steel blade, while the hammer was made quite a bit longer with a thick handle and large cast iron head. The sword has a single handle point, as it is only meant to be carried with one hand, while the hammer has two and can thus only be held correctly using both controllers. The hammer's handle contains two large leather grips, further indicating the need to carry it with both hands as shown in Figure 6.

5.4 Weapon Impact

With the purpose of having something to test the weapons on, we created an opponent who reacts to weapon impacts. The viking opponent contains several animated colliders. To give a visual indication of the force behind the objects being swung, several different reaction animations were created for the viking opponent: an idle stance, struck from the left, struck from the right, knocked back, and knocked down. Transitions back to the idle state are used when an animation has completed and none of the other possible transitions are active.

The viking enters the idle state at the beginning of the scene and awaits collision with other game objects. Upon collision, the force and direction of impact are used to calculate the next state. If the force is low enough nothing will happen, but if it exceeds the minimum amount, the viking will pull back his shoulder and turn his upper body. If he is struck with a greater amount of force a knock-back animation will trigger, where the viking takes a step back while bending his upper body backward. Finally, if he is struck very hard with a blunt object like the hammer, the viking opponent will fall backwards onto the ground.

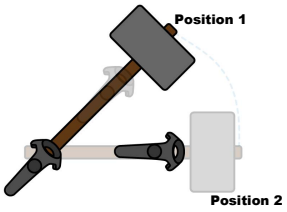


Figure 7: Hammer chasing the controller during rotation. Rotation is delayed when user motion is too quick, so that the hammer needs a brief moment of slower user motion to catch up with the controller closest to the hammer head.

6 WEIGHT

Based on experiments regarding object handling and weight, our main objective is to investigate whether software-based interaction methods can make a user interact with virtual objects more realistically. Given the test scene, its grabbing mechanics and its affordances, we developed three methods to better indicate the weight of an object. We refer to two of these methods as *chasing* and *scooping*. As a third method, we added haptic feedback from the controller during specific interactions.

6.1 Chasing

Chasing was chosen as the name for our method of making the hammer, or potentially other heavy objects, "chase" the controller during the interaction as a way of indicating its heaviness. The hypothesis is that the hammer might seem heavier, as it acts more slowly, and spurious movement is limited.

We use linear interpolation of quaternions (deliberately not spherical linear interpolation) to make the hammer chase the controller in the desired manner. The hammer only chases the controller near the hammer head, as this is where the rotation takes place. The other controller determines translation, as the centre of mass is not close to this point, and the user should still feel in control of the hammer. We find that this gives the desired indication of heaviness. Figure 7 illustrates an example of the chasing functionality where the rightmost controller has moved from position 1 to 2, but the hammer is (for the moment) still waiting at position 1. A couple of frames later, the hammer will have moved to position 2. The interpolation is greatly exaggerated, as the hammer will still be much closer to the controller at most times. The leftmost controller in Figure 7 is unchanged, indicating that the translation is still one-to-one.

From frame to frame, we use the difference between the new and old velocity of the hammer as an acceleration measurement. If the acceleration factor is much too high, the object will be released (dropped), making the user lose control of the hammer completely. If the acceleration factor is slightly too high, the interpolation

Figure 8: Handling of balls after the implementation of Scooping.

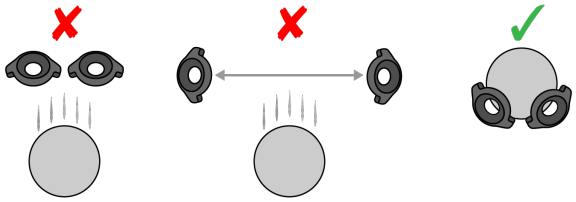


Figure 9: From left to right: Hands too close to the top, hands too far apart, Scooping is done correctly.

is set to be very slow, which forces the user to move the controller back to the correct position on the hammer before doing another swing, or wait for the interpolation to finish. However, the increase of the interpolation factor will accelerate with 5% for each frame, to ensure that the user needs not wait too long. If the hammer is swung with a realistic acceleration factor, it will move normally.

6.2 Scooping

Scooping is our method for handling heavy objects that we want to lift using two hands. When scooping, the user makes a gesture that takes the form of a bowl or a shovel. This can be used to "scoop" certain objects. Figure 8 illustrates the scooping gesture.

We implemented scooping for a set of test balls. The idea behind scooping is that many people would not be able to hold a heavy steel ball with a clawed fist, therefore they should not be able to do so in the virtual environment. However, many people would still be able to lift the same steel ball using two hands. If the hands are both directly under the ball applying force in the parallel opposite direction of gravity, the user would have the strongest grip. Sliding both hands towards the top of the ball would make the applied force smaller. Hence, the grip would become weaker when closer to the top, and eventually we would drop the ball. If the user was to move his hands too far apart, the ball would also be dropped. This is illustrated in Figure 9.

As opposed to our other types of interaction, scooping is done by using a grip button on the side of the HTC Vive controller instead of pressing the trigger. This felt more natural. When the controllers collide with a ball while gripping, we check if the object has any handle points. If not, scooping commences. The ball then centers itself between the controllers. We can then compute the positions and rotations of the controllers.

Regardless of the controller rotations, if the distance of the controllers to the ball becomes larger than the ball radius plus a threshold, the object is released and the ball is then dropped. This threshold has been chosen so that it corresponds approximately to a 5 cm real life distance to the virtual ball from each controller.

The virtual weight of the test balls are in the interval $[0.7, 1.9]$. We therefore set up the following formula to estimate the weight w that the user can lift when making the scooping gesture:

$$w = 2 - \frac{\cos(\theta_{\text{left}})\cos(\phi_{\text{left}}) + \cos(\theta_{\text{right}})\cos(\phi_{\text{right}})}{2}.$$

Here θ and ϕ are angles of controller rotation with respect to the two horizontal world space axes. The subscript indicates the left or the right controller. If the user does not rotate the controllers at all (zero angles, Figure 9, left), which corresponds to having the palms turned downwards, the calculated weight that can be lifted is $w = 1$, making it the weakest grip. Conversely, turning the hands 180° to have the palms pointing upwards ($\theta_{\text{left}} = \theta_{\text{right}} = \pi$, Figures 8 and 9, right) results in $w = 3$, making it the strongest grip. If the calculated weight is less than the virtual weight of the object, the object is dropped. For example, if neither of the controllers are rotated ($w = 1$), a light ball of fabric with the weigth of 1 may still be scooped, but a wooden ball with a weight of 1.2 would be dropped. By doing the implementation with this approach, a heavy steel ball will be dropped much faster than a wooden ball, as it should. The selected virtual weights of different test balls are in Table 1.

6.3 Haptics

Haptic feedback can for example be used as an indication of wind resistance and strain. We therefore use haptics when interacting with most objects, as well as for the chasing and the scooping functionalities. A force parameter controls the haptic feedback in a controller, so we set a force based on the object velocity whenever the user is swinging a weapon. This creates a good indication of wind resistance as the weapon slices or crashes through the air. When the hammer is chasing the controller, we apply the maximum haptic force, giving the indication of swinging the hammer in a wrong manner and feeling the strain of doing so.

We use the haptic feedback slightly differently when scooping. Here the haptics are used as a warning of when the ball is close to being dropped. This also corresponds to the strain one might feel, when struggling with carrying a heavy object, because it is held in a wrong way. The haptic force will be in the range from zero to maximum depending on the difference between the lifted weight w and the virtual weight of the object. Haptic feedback (vibration) starts at an angle of approximately 35° before dropping the ball.

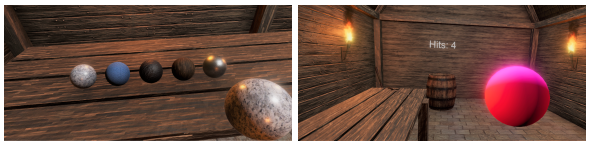


Figure 10: Test balls for scooping test.

Material	Granite	Fabric	Cast Iron	Wood	Steel
Mass	1.4	1.0	1.6	1.2	1.6

Table 1: Materials and associated virtual masses of the test balls. A purple ball is of undefined material and has a mass in $[0.7, 1.9]$.

7 TESTS AND RESULTS

Two test sessions were conducted: one without chasing and scooping and one with chasing and scooping. The first test had nine participants (three females and six males), the second test had eleven participants (three females and eight males). Seven test subjects participated in both tests and thus had some experience with the virtual environment when testing with chasing and scooping. None of the test subjects had any previous experience with virtual reality, but they all had an amount of experience with computer gaming. The test subjects were 19 to 26 years old and got as little help from us as possible during the tests. We got permission from everyone to film them, as well as their permission to use their test results in written work.

7.1 Different Virtual Masses

In many VR systems, a wooden table or a massive stone block can be thrown equally far. This can be fun, but an object not accounting for weight in its affordances is not very realistic. We set off one part of our test session for investigating whether realistic textures and different object sizes give an impression of the weight of an object. Figure 10 shows an overview of the test.

In the test, an operator would generate a ball by pressing a button. The system randomly selects one of six different ball types, and the test subject attempts to throw the generated ball into a barrel. In a typical test, the subject threw around 50 balls. The materials and masses of the balls in Figure 10 are listed in Table 1. The virtual mass is measured in arbitrary units. We refer to the ball with no texture as the "purple ball". The purple ball was assigned a random mass between 0.7 and 1.9, and its radius was scaled accordingly.

The test subject is presented with a ball, picks it up (with or without scooping functionality) and throws it toward the barrel from a marked position by the table. Every time the ball collides with the bottom of the barrel, a "Hits"-sign increases by one. The number of hits are not important, but the test subject got a little game out of the experience (trying to have as many hits as possible), which adds to the enjoyment. This should lead to more focused throws and thereby better test data.

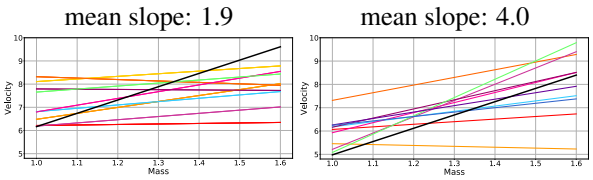


Figure 11: Plots of controller velocity by mass regression lines in ball throwing test without scooping (left) and with scooping (right). Each user has a separate color. Black user knows the masses.

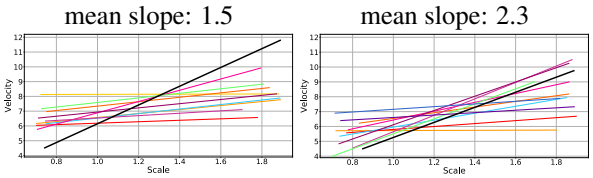


Figure 12: Plots of controller velocity by scale regression lines in ball throwing test without scooping (left) and with scooping (right). Each user has a separate color. Black user knows the masses.

The data logged in this test was the ball type, the controller velocity when the ball is released, and the mass of the ball. After filtering out probable outliers, we apply a linear regression to the data points. Outliers are typically the balls that were dropped by accident.

Going from lighter balls to heavier balls, the test should ideally show a positive regression slope. This seems counter-intuitive because it is unphysical, but larger controller velocity upon release becomes the same ball velocity in the virtual reality (and the same ball velocity is desired if the user wants to hit the barrel and score a point). The extra force applied by the user when moving the controller faster compensates for the fact that the controller weighs the same for all objects.

7.1.1 Velocity Graphs

Plots of controller velocity by mass and by scale are in Figures 11 and 12, respectively, where the "ideal" regressions shown in black are the results of testing ourselves, already knowing the mass of each ball.

In Figure 11 (left), we see that use of textures and division of the ball velocity upon release by the mass when released (as explained previously) gives an indication of the weight of the object. Subjects have a general tendency to throw the heavier balls with a higher controller velocity, albeit not a very strong one. A few of our test participants on average threw all of the balls with almost the same amount of force, regardless of their apparent weight. The regression lines of those test subjects still have some of the lowest slopes when our scooping method is employed (Figure 11, right). However, we generally see a stronger tendency to throw the balls with a force corresponding to their masses when scooping is employed. Lighter balls were thrown with a lower controller velocity than before, while heavier

Figure 13: Example of test-fight using the hammer.

balls were still thrown at a high velocity. Some of the test subjects even exceeded the "ideal" results achieved by ourselves, leading to the conclusion that the scooping method accompanied by material textures is a very effective method of communicating weight to the user.

The balls of varying size with no material texture were less effective. The difference in throwing force from the smallest to the largest ball was generally small. While a few test subjects came close to the ideal regression in Figure 12, most lines have a much smaller slope. Once again the scooping method made most of our test subjects throw the balls in greater accordance with their virtual masses.

Inspecting the average controller velocity for each material, we observe in tests without scooping that granite balls were thrown harder than steel balls, and that iron balls were thrown harder than steel and granite balls. This has had an impact on the regressions as the steel and iron balls had the same mass implemented, which is higher than that of the granite ball. Furthermore, the fabric balls were thrown at a rather high controller velocity, even though they were the lightest. Test subjects may have anticipated a wind resistance, but this was not implemented in the application.

When using our scooping method, the mean slopes increased significantly. The combination of scooping and material textures moved the average throw much closer to the ideal (slope 6.3). Altogether, both with and without scooping, the material textures seem to have communicated the weight of the balls a lot better than the size differences. The purple colour used for the scaled balls may have been misinterpreted by some as a light material (e.g. a rubber balloon).

7.2 Swinging with One or Two Hands

In another part of our test session, we compared the swing of the sword and the hammer. In the first test session, we let the user swing the hammer in the same manner as the sword, only using two hands instead of one. We wanted to test whether the hammer would be swung more realistically due to its heavier appearance. In the second test session, our chasing method was employed to further indicate heaviness, while the sword

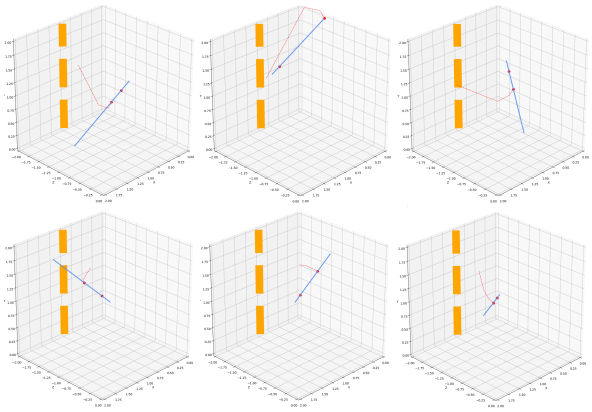


Figure 14: Hammer swings of two test subjects (one tester in each row).

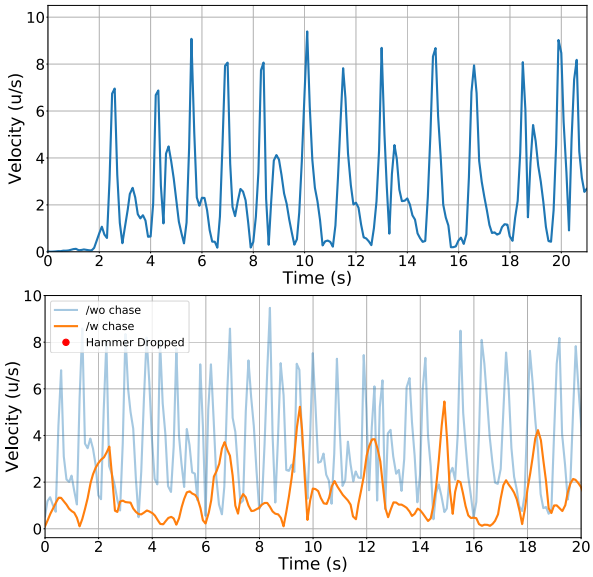


Figure 15: Examples of swing frequency plots for the sword (top row) and the hammer (bottom row).

would still follow the controllers one-to-one. Figure 13 shows an in-action example of a subject fighting the viking with the hammer.

Each test subject would fight the viking for about 40 seconds with both the sword and the hammer. We recorded each session and logged time, weapon velocity, and position and rotation of the controllers. These values were logged once every 10 milliseconds.

From the logged data we created two types of plots. The first type is an interactive 3D graph, that enables us to investigate the weapon swings from different angles and view the position of the two controllers throughout the swing, see Figure 14. The second type of plot shows the velocity of the weapon over time, see Figure 15. We use these plots to assess whether a test subject is using the hammer as one might in reality.

We investigate the hammer orientation during a swing to see whether the test subjects were immersed enough to hold the hammer in the same way as they would in

real life. We also investigate the sword and hammer swing frequency to see whether a test subject would swing the virtual weapon realistically or uncontrolled and vigorously.

7.2.1 Hammer Orientation

Logging the position of the controllers while the hammer is swung enables us to analyze both the path of each swing and where on the hammer handle the test subject would place the hands during a swing. The graphs produced from the data show the controller positions as red dots and the hammer as a blue line between them. The path of the swing during the past 300 ms is shown as a red dashed line and the opponent that the test subjects were told to hit is shown as an orange dashed line.

Figure 14 (top row) shows a couple of hammer swings from a test subject. From left to right, he swings the hammer downward into the opponent, lifts it up above his head, and swings it down again. As seen in the plot, this test subject would typically hold his hand near the head of the hammer when preparing for a swing and then move his hands closer together during the swing. These hand positions correspond well to the way one would swing a sledgehammer in real life, as you would want your hand closest to the center of mass when lifting it above your head, but slide it toward the bottom of the handle when swinging, allowing the sledgehammer to pivot about your hands.

In our implementation, since the hand at the bottom of the hammer controls its position and the other hand controls its orientation, the controllers do not need to be an equal distance apart throughout the swing. This allows for swings such as the one explained above, but also for ones that would not be possible in the real world.

Some of our testers would hold one controller statically in place and use the other to rotate the hammer about that point as shown in Figure 14 (bottom row). While this would seem unrealistic to do with a real sledgehammer, it was less noticeable inside the virtual environment. Perhaps enforcing an equal distance between controllers when holding the hammer or making the user hold the controllers at the same angle as the hammer would have yielded more realistic results among more of our test subjects, but adding too many restrictions could also greatly reduce their immersion.

7.2.2 Sword and Hammer Swing Frequency

In the test without chasing, the sword and the hammer were handled in the same way. The only difference was the heavier appearance of the hammer and the two handle points. We may contemplate based on Figure 15 whether the appearance and extra handle point of the hammer was enough for people to swing it differently from when swinging the sword.

In the hammer swing frequency plot (Figure 15, bottom row), the orange curve is from the test with chasing, whereas the faded blue is without chasing. Red dots indicate if the hammer has been dropped from swinging it too violently, which has not happened in this plot. Additional plots are available in a supplementary document. Comparing the sword swing frequency plot with the one for the hammer without chasing, there is no noticeable difference, meaning that the sword and hammer were handled roughly in the same way. This corresponds to feedback in our TPI tests, where people answered that the sword and hammer weigh approximately the same. If the test subjects had actually been tricked by the size and texture of the hammer, the swing of the hammer would generally have had a lower velocity and a greater wavelength compared to the sword. Interestingly, this is exactly the case after the implementation of chasing. The orange curve has a much lower velocity in general, and a greater wavelength compared to the blue curve. This is due to the fact that our chasing method forces the user to stabilize and swing the hammer in a realistic manner.

Although the data used in Figure 15 is from one test subject only, most other subjects revealed the same trend in their versions of this plot (see supplement). However, a few test subjects performed roughly the same both with and without chasing. These subjects were possibly also the ones not fully immersed in the experience, making them move the hammer very slowly and carefully around in both test sessions.

To confirm that the orange curve in Figure 15 is indeed a more realistic two-handed interaction with a long, heavy object, we conducted a test in a different virtual environment. This is described in the following section.

7.3 Baseball Bat Test

In this test, we use substitutional reality to investigate the differences between using a standard VR controller and a real life object. We use a baseball hitting simulation to measure the speed of a virtual bat's barrel when trying to hit a baseball. The real bat is tracked via a controller attached to it (see Figure 16), but can be freely interacted with as a normal bat.

In the simulation, the speed of the virtual bat's barrel in global space is constantly recorded so that the speed of the swing attempts can be measured. As seen in Figure 17, real bat swings cannot reach as high a maximum speed as using a controller. The real bat also requires more time to accelerate and decelerate between rest and swings. This is likely because a lightweight controller in the center of one's grasp requires much less effort to manipulate due to its weight and center of mass. Perhaps this absence of strain is the reason why some users, after conducting both tests, would surprisingly prefer the unmodified controller to the real bat.

Figure 16: An Oculus Touch controller attached to the base of an aluminum baseball bat. We use this aggregate in a test conducted in a crude stadium scene with a pitching machine.

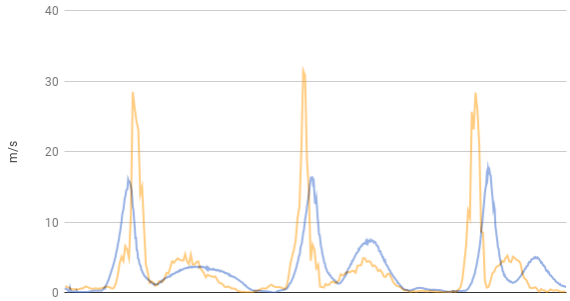


Figure 17: Virtual bat swing speeds using a controller (orange) and a real baseball bat (blue). The high spikes represent attempts to hit a virtual baseball.

We can now compare Figure 17 with the test results in Figure 15. Remarkably, our chasing technique seems to have qualitatively the same effect as if the user switches from using an unmodified controller to using a heavy, real object. The change to lower velocities and more stabilized motion between the hits is strikingly similar in the two figures. This is encouraging as we wanted to simulate realistic tangible mapping.

7.4 Temple Presence Inventory (TPI)

The TPI mentioned introductorily is a different kind of test. It is a questionnaire, where the test subject answers questions on a scale from 1 to 7. In our TPI, the questions are separated into five categories: ball interaction, weapon interaction, spatial presence, engagement, and perceptual realism. We provide a full overview of our TPI questions and results in a supplementary document.

The ball interaction category contains questions such as "how real did the wood ball feel regarding weight, size and overall texture?" And "how real did it feel to throw the balls?" The purpose of this approach is to assess what the test subject feels with respect to throwing the test balls. The weapon interaction category will ask if the hammer looks heavier than the sword, and whether or not it feels heavier to swing. The performance tests discussed previously were designed to assess the realism of user performance. In these ball and weapon interaction categories of the TPI, we ask the user what they felt about the interactions.

The last three categories: spatial presence, engagement, and perceptual realism together determine, each in their own way, how the user experienced the virtual environment. As mentioned previously, we believe the user will only expect the affordances of the objects to be realistic if the environment itself looks and feels realistic. The test subjects were therefore first free to walk around the virtual environment. They got to know the scene by lighting torches and trying the shield and an axe. We wanted to avoid low scores in these categories. Low scores here *and* poor performance tests would indicate a problem in the way the environment is portrayed rather than in the interaction design.

7.5 TPI questionnaire

The first section of the TPI was on the *ball interaction test*, asking how real the balls felt both individually and all together. In the test with scooping, subjects found that wood and iron balls felt more realistic. However, everything else got a worse result or nearly the same result as in the session without scooping. For steel and granite balls, which got lower scores with scooping, one explanation could be that the test subjects became frustrated with the controller rotations required to scoop those balls, while the wood ball weighed much less and was thus more forgiving. Hence, some balls seemingly got a worse score simply because they were more difficult and not as fun to throw. In any case, the test scores (with or without scooping) are quite high.

The next section of the TPI is the *weapon interaction test*. From this questionnaire, we observe that with our chasing method the hammer felt approximately 23% worse to swing. However, when asked about its heaviness compared to the sword, the score was about 36% larger than in the test session without chasing. Other answers indicate that the hammer does not feel quite as heavy as it looks, but it got closer in the test with chasing, where subjects found it much heavier than the sword. The conclusion resembles the one from the ball interaction test: the hammer was more realistic to swing and felt heavier, making it more difficult to use, and perhaps it was then not as much fun to swing it.

The sections on *spatial presence*, *perceptual realism*, and *engagement* can be grouped together into an overall experimental measurement of immersion. The scores are very high with the average spatial presence of the

participants being 6. We believe initial interaction with torches and items like the shield were important in ensuring this high number. For perceptual realism, the average score was around 4.5, which is also great as these specific questions are slightly better suited for substitutional reality. The motion sickness score is also very low, which means that the users did not experience much lag, unsteady locomotion, or light flickering.

TPI answers indicate that test subjects went from a score of 5.3 to a score of 5.9 with respect to feeling mentally immersed in the experience when scooping and chasing were included in the tests. In addition, the participants were a bit more relaxed during these tests and less engaged regarding all their senses. However, the difference between the scores is not substantial. We may conjecture that the experience was perhaps slightly more relaxing as it was no longer possible to interact with everything as vigorously as before the use of chasing and scooping. The average score was about 5.5 in both test sessions, which is quite high.

From the TPI scores, we conclude that the experience of the virtual environment was altogether both realistic and immersive.

8 CONCLUSION

We sought to improve upon the realism of interactions in virtual reality by using software solutions to encourage natural handling of objects. A virtual environment for the study was created in the form of a room populated with various objects of differing size, shape, and functionality. Basic mechanics for grabbing and moving these objects were implemented followed by interaction possibilities corresponding to their fundamental affordances. More advanced interaction techniques depending on the mass and shape of objects were then implemented in the form of a chasing method and a scooping method.

Performance tests regarding throwing and swinging mechanics were conducted both with and without use of the chasing and scooping methods. We used a presence questionnaire to assess the user feeling with respect to interaction realism and immersion.

The test subjects found themselves highly immersed in the virtual environment, but interacting quite unrealistically with the test objects when our interaction techniques were not employed. Use of our methods led to a general improvement in the realism of object handling and the feeling of weight. Nevertheless, this led to only a small improvement in immersion. Test subjects also implied that the interactions were overall less enjoyable with our techniques. Their levels of enjoyment could perhaps have been maintained, had the implementations been a bit more forgiving.

In conclusion, it was possible to achieve a feeling of weight through the proposed methods of object handling, thereby simulating realistic tangible mapping.

9 REFERENCES

[AHB*16] Azmandian M., Hancock M., Benko H., Ofek E., Wilson A. D. Haptic retargeting: Dynamic repurposing of passive haptics for enhanced virtual reality experiences. In *Proceedings of Conference on Human Factors in Computing Systems (CHI)* (2016), pp. 1968–1979.

[CCM*17] Choi I., Culbertson H., Miller M. R., Olwal A., Follmer S. Grabity: A wearable haptic interface for simulating weight and grasping in virtual reality. In *Proceedings of ACM Symposium on User Interface Software and Technology (UIST)* (2017), pp. 119–130.

[DLB*05] Dominjon L., Lécuyer A., Burkhardt J.-M., Richard P., Richir S. Influence of control/display ratio on the perception of mass of manipulated objects in virtual environments. In *Proceedings of IEEE Virtual Reality (VR)* (2005), pp. 19–25.

[JAO*14] Jáuregui D. A. G., Argelaguet F., Olivier A.-H., Marchal M., Multon F., Lécuyer A. Toward "pseudo-haptic avatars": Modifying the visual animation of self-avatar can simulate the perception of weight lifting. *IEEE transactions on visualization and computer graphics 20*, 4 (2014), 654–661.

[Jer16] Jerald J. *The VR Book: Human-Centered Design for Virtual Reality*. ACM and Morgan & Claypool, New York, NY, USA, 2016.

[LDW09] Lombard M., Ditton T. B., Weinstein L. Measuring presence: the temple presence inventory. In *Proceedings of the 12th Annual International Workshop on Presence* (2009), pp. 1–15.

[MFK*07] Minamizawa K., Fukamachi S., Kajimoto H., Kawakami N., Tachi S. Gravity grabber: Wearable haptic display to present virtual mass sensation. In *Proceedings of ACM SIGGRAPH 2007 Emerging Technologies* (2007), p. Article 8.

[RGGR18] Rietzler M., Geiselhart F., Gugenheimer J., Rukzio E. Breaking the tracking: Enabling weight perception using perceivable tracking offsets. In *Proceedings of Conference on Human Factors in Computing Systems (CHI)* (2018), ACM, p. Paper 128.

[SCP11] Shafer D. M., Carbonara C. P., Popova L. Spatial presence and perceived reality as predictors of motion-based video game enjoyment. *Presence: Teleoperators and Virtual Environments 20*, 6 (2011), 591–619.

[STS*11] Skalski P., Tamborini R., Shelton A., Buncher M., Lindmark P. Mapping the road to fun: Natural video game controllers, presence, and game enjoyment. *New Media & Society 13*, 2 (2011), 224–242.

[SVG15] Simeone A. L., Velloso E., Gellersen H. Substitutional reality: Using the physical environment to design virtual reality experiences. In *Proceedings of Human Factors in Computing Systems (CHI)* (2015), ACM, pp. 3307–3316.

[ZK17] Zenner A., Krüger A. Shifty: A weight-shifting dynamic passive haptic proxy to enhance object perception in virtual reality. *IEEE Transactions on Visualization and Computer Graphics 23*, 4 (2017), 1285–1294.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

3D Annotations for Geospatial Decision Support Systems

Silvana Zechmeister
VRVis Research Center
Donau-City-Straße 11
Austria 1220, Vienna
silvana.zechmeister
@student.tuwien.ac.at

Daniel Cornel
VRVis Research Center
Donau-City-Straße 11
Austria 1220, Vienna
cornel@vrvis.at

Jürgen Waser
VRVis Research Center
Donau-City-Straße 11
Austria 1220, Vienna
jwaser@vrvis.at

ABSTRACT

In virtual 3D environments, it is easy to lose orientation while navigating or changing the view with zooming and panning operations. In the real world, annotated maps are an established tool to orient oneself in large and unknown environments. The use of annotations and landmarks in traditional maps can also be transferred to virtual environments. But occlusions by three-dimensional structures have to be taken into account as well as performance considerations for an interactive real-time application. Furthermore, annotations should be discreetly integrated into the existing 3D environment and not distract the viewer's attention from more important features. In this paper, we present an implementation of automatic annotations based on open data to improve the spatial orientation in the highly interactive and dynamic decision support system Visdom. We distinguish between line and area labels for object-specific labeling, which facilitates a direct association of the labels with their corresponding objects or regions. The final algorithm provides clearly visible, easily readable and dynamically adapting annotations with continuous levels of detail integrated into an interactive real-time application.

Keywords

Automated Label Placement, Map Annotation, Geospatial Visualization, Open Data

1 INTRODUCTION

The economic growth and climate change have a great impact on the frequency and intensity of natural disasters worldwide. River floodings affect many people and cause a lot of damage and costs. Therefore, the need for flood management systems to analyze different flood scenarios grows, especially in densely populated areas with river proximity.

Visdom is a flood management system which supports interactive decision making based on fast geospatial simulation and visualization. It offers the opportunity to test different protection measures, to be prepared for flood disasters and to act correctly in serious situations. A good support of spatial orientation and navigation in urban 3D environments for flood managers, relief workers and other persons involved is needed. A common approach to achieve this aim is the use of annotations. Thus, this paper covers the extension of Visdom with labels for different landmarks to know where the affected areas are located and to stay oriented while navigating through the system.

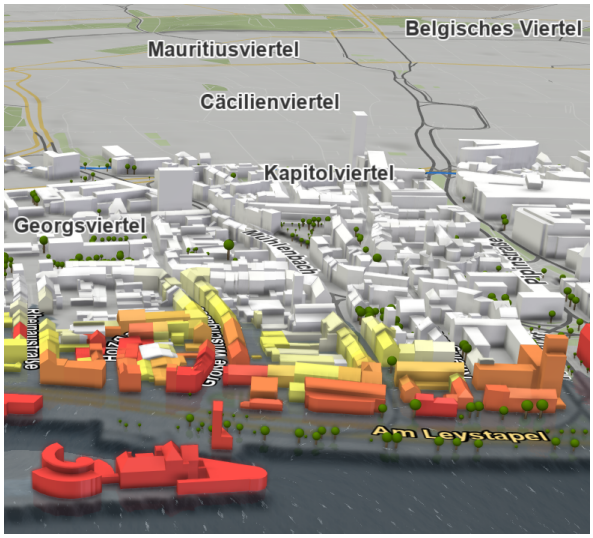


Figure 1: Integration of area and line labels into an interactively explorable and dynamically changing 3D environment with rising water.

The integration of annotations in a 3D dynamic environment causes different challenges compared with static 2D city maps. In contrast to static approaches, dynamic systems need a good trade-off between computation time and optimal label placement to enable a real-time user interaction. In Visdom, the annotations

should not impede the user by visual distraction or by hiding important simulation data. The major goal is not to place as many labels as possible but to support the navigation in a subtle way. These requirements increase the importance of a good visual integration into the dynamically changing 3D environment (see Figure 1).

The label data used for the annotations are from Geofabrik [Geo18], a free download server which extracts geospatial data from OpenStreetMap [Ope19] and provides map textures, line and area shape files. The textures do not allow the partitioning of label data and thus updating comes along with loading high volumes of data and long waiting times. This impedes dynamic updates during runtime, while the label resolution, orientation and size have to stay static. Their inflexibility is not applicable with the high interactivity and large zoom range of Visdom. Furthermore, texture based labels mapped to the ground are unsuitable for annotating 3D objects such as buildings because they are hiding their own labels. To guarantee label readability, the labels should be embedded into the 3D world and their font size and orientation should dynamically adapt to the zoom factor and the point of view.

OpenStreetMap offers an extensive amount of line and area segments with their own labels and partially incomplete label text. The high segmentation of lines results in too many labels in the most cases. To produce appropriate annotations, preprocessing and intelligent label placement has to be executed in advance. When placing a label along a line, the positioning of its letters is not trivial. They have to follow arbitrary curves which can have a negative impact on label readability. Thus, the aim is a label appearance that is as straight as possible. To make the depth information more perceptible, the annotations should be occluded by other scene objects. The overlap between two labels should be avoided, since this would destroy their legibility. It is essential in a real-time application such as Visdom, to efficiently test labels for intersection with others.

This paper presents an implementation of annotations which solves the mentioned problems and eases the localization and identification of streets, rivers, buildings, places, city districts and other landmarks.

2 RELATED WORK

There is a wide area of application for labeling features, it is used in cartography, geographic information systems and point pattern analysis for instance. Thus, there exist various techniques to find a label placement and layout according to the annotation application.

Label Placement

Marks and Shieber [MS91] show that label placement on a map is an NP-hard problem and it needs heuristics to label large quantities of features. To find best possible label positions, the definition of the optimum is essential. Van Dijk [DKSW02] reviews existing label placement rules and defines four categories to evaluate label placement quality. The four quality criteria of Van Dijk are aesthetics, label visibility, feature visibility and label-feature association. The most common label features are points, lines and areas to annotate cities, streets and countries for instance.

Research in the area of 3D maps deals with automating and optimizing the annotation process concerning the quality and quantity of label placement. In the case of dynamic environments with pan and zoom operations, interactive frame rates with legible annotations embedded into the 3D scene are additional challenges. Niedermann and Nöllenburg [NN16] provide a framework that labels 31% more road sections than the standard OpenStreetMap renderer with near-optimal quality based on quality criteria similar to Van Dijk's. But their algorithm has running times in the range of seconds to one minute which does not support dynamic user interaction. Schwartges et al. [SWH14] propose an algorithm that produces high quantities of aesthetically pleasing labels but it has also problems to provide real-time interaction due to slow label placement during zooming. The mentioned approaches project the annotations onto planes because of the easy implementation and good readability. This impedes a direct label integration into features with irregular surfaces such as terrains and may result in low label-feature association. Maas and Döllner [MD08] use parametric hulls to embed labels also in curved surfaces to label buildings. Their approach is limited to three different shape types, which does not allow the labeling of streets on uneven terrains. She et al. [SLL+17] use line features as label base lines to place labels on arbitrary terrains. Labels nearly perpendicular to the screen become indecipherable. In this case they are not placed along their corresponding lines, which impairs their relationship.

Text Rendering

The use of signed distance fields is a way of text rendering that tries to overcome the resolution limitation of raster graphics [Gre07]. The signed distance field texture can be very small and it still produces crisp text. But this approach does not accurately represent text contours near to complex letter intersections and it is not appropriate for very small text sizes [Gus12]. In such cases, the use of direct rasterized vector graphics may produce better results [Rou13].

Visualization Techniques

Depending on its appearance, a piece of text can attract attention, be seamlessly integrated into the environment or increase legibility and the appealing look of the whole 3D scene.

Annotations are categorized by the visualization technique into external and internal labels. External labels avoid hiding their belonging feature by placing them besides the feature and connect them with anchor elements for their association. Their main application area is the annotation of single 3D objects such as scientific illustrations [TKGS14]. Multiple external labeled objects lead to worse label-feature association and visual clutter. Internal labels, which are directly embedded into its 3D feature are more suitable for virtual environments with many features.

Adding an outline to letters can increase their contrast to the background. The use of halos around labels, which clear the space close to them, leads to further readability improvement by losing environment information [VTW12]. Vaaraniemi et al. [VFW13] introduce methods to enhance the visibility of labels by fading, cutting and removing other scene objects that occlude the labels and their features. These methods are also at the expense of environmental information.
The dynamic change of label position and orientation enables the adaption to user interaction for better visibility and readability. But such behavior can result in flickering and thus distract the user. To avoid this effect, Maass and Döllner use label blending and animation [MD07]. With blending, the labels smoothly fade in before appearing and fade out before disappearing. The animation is used to continuously move a label from one position to another.

3 PREPROCESSING

The first step of the annotation pipeline is the preprocessing of input data provided by OpenStreetMap [Ope19]. The data need to be prepared to achieve an efficient further use and to enable fast rendering. The incoming data are labels and two different shape types. There are optional importance and color values available to adapt the label output accordingly. A label is assigned to a certain shape and a position on or near this shape. Its corresponding text might be missing or empty, depending on the OpenStreetMap data. In this case the label is not taken into account. The two different input shapes are lines and 2D polygons, which are used for streets, places, buildings and other landmarks. A line is defined by a number of control points. The connection of all points results in the line to which the control points belong. A polygon is defined by a set of consecutive vertices, whereby the first and the last vertex are the same.

The OpenStreetMap data include numerous line segments and each segment has its own label. If we would render all these labels, it would result in an overloaded and confusing scene. To avoid this, we want to merge lines and their corresponding labels together. Our approach is to merge two line segments together if they have the same label text and are adjacent to each other. When merging line segments, the labels assigned to them are merged too. This is done by setting the merged label into the middle of the new merged line segment. But some restrictions are needed for the line merge process to prevent merging lines when their corresponding objects (e.g. streets) cannot be clearly identified. This is the case if it is not obvious which direction a line course is following. To avoid such a scenario we check the angle between the potential end positions to be merged and if it is acute-angled, we do not merge. This results in easily identifiable landmarks that are labeled before and after strong bends. The polygons have at most one label, so there is no need for a reduction as for the lines.

The next preprocessing step after the line merge is the label sorting. Labels are sorted according to their camera distance and their optional importance values. The importance values are used to improve the final output. More important labels are visually differentiated compared with less important labels and get a preferential treatment. More relevant labels are rendered first to increase the probability of their visibility. But independent of the importance values, labels near the camera should be paid particular attention, since these are most likely labels the viewer is interested in. Thus, labels with the same importance are treated according to their distance to the camera. The camera distance changes more frequently than the importance values because it depends on the navigation through the environment. The labels have to be sorted by their camera distance each frame, but their order according to their importance stays the same and can be precomputed during the preprocessing.

For the application of rendering label text in Visdom we decided to use vector graphic fonts that allow an adaptive rasterization for all required font sizes. In the field of typography, characters are represented by glyphs and a vector graphic font contains accessible glyph metrics. With a given label position, each letter can be placed appropriately by the use of its metrics. All labels are rendered with the same font, only in different size and therefore its glyph metrics can be loaded and stored for later use. Furthermore, texture atlases containing the most common 256 letters and their outlines are generated through rasterization of the letter's vector graphic descriptions. There are several texture atlases needed to cover all required font sizes. If only one texture atlas would be used and scaled to fit the appropriate font size, the letters would have a poor, upscaled resolution. Thus, texture atlases for all initial font sizes are generated in the preprocessing step.

The last preprocessing step concerns the label orientation and its great impact on readability. Most languages have a write direction from left to right and therefore one is trained to read in the same direction. This holds

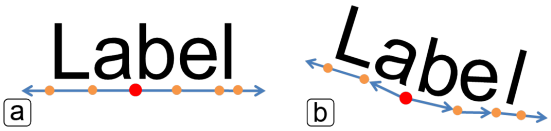


Figure 2: Illustration of label (red) and letter positions (orange) on a horizontally placed area label (a) and a line label placed along its corresponding line (b).

true for the annotations in Visdom, which are mostly in German. This is why we aim to orient the labels from left to right along the line segments to ease reading. The order of line control points determines the orientation of the corresponding line label. To adapt the label orientation appropriately, the control point order needs to be dynamically changed according to the current view. For a quick access during rendering, the original and the reversed order of control points are stored.
With this prepared and quickly accessible information, text rendering can be executed in a fast and efficient way.

4 LABEL & LETTER PLACEMENT

After processing the input data, the labels and their individual letters need to be placed at the right position. The label placement is a very complex task with several possible solutions. Some of them are already mentioned in section 2. The main criterion for label and letter placement in Visdom is the execution time. It has to be very fast to be able to afford real-time interaction. But a good placement has to adapt to the current view to improve its readability by changing its scale and orientation. As a result, label placement is view-dependent and is therefore changing very often and needs plenty of time to process. Due to these preconditions, the aim is not to find an optimal label placement but a satisfactory and fast placement.
The use of fixed instead of dynamic label positions turned out as a good choice. It avoids flickering and saves important runtime through skipping dynamic calculations of label positions. After the determination of a line or area label's position, each character has to be placed along a line or above a point of interest. Contrary to the determination of the label position, the letter placement is executed dynamically to change its orientation and scale according the current view. By means of the glyph metrics, the letters can be placed in the scene with their positions only. Thus, the aim is to get the positions for all letters.

4.1 Area Labels

To determine the letter positions it is necessary to get the floating label positions first. This is accomplished by shifting up the area's midpoint. The label position (see Figure 2a, red) is then the position of the middle letter too. Starting from this point each letter is placed horizontally to the screen by using its corresponding advance. The glyph advance is the horizontal space needed to place a certain letter accurately and can be extracted from the glyph metrics. The letter positions are calculated by subtracting the glyph advance if the letter is to the left of the label position and adding it otherwise. The orange points of Figure 2a represent the positions of the letters and the space between them is referred to as the advance.

The final output is a label that is horizontally centered and floating above its corresponding area. Caused by using the screen-space for the placement, area labels have a constant size and are always facing the viewer. This means they have a high readability because the text is not scaled when zooming in or out, has no distortion and is always orthogonal to the screen.

4.2 Line Labels

In consequence of fixed label positions, every line label is permanently located at the middle of its line segment. A usual practice of line label placement is the projection of the labels onto their corresponding objects to increase the label-feature association. But to achieve this, the letter placement needs to be performed in world-space. The constant size of area labels turns out to be handy because the labels are never over- or under-dimensioned and are always easily readable. To get such a constant label size also for world-space placement, a scale factor is used, to resize letters according the current zoom factor. The field of view and the distance between the label and the camera is taken into account to get an appropriate scale factor. This factor enables a nearly constant label size with respect to the screen and thus facilitates better label readability.

The letter placement starts at the label position, which is the red point in Figure 2b. Moving along the line by the same glyph advance procedure as for the area labels provides letter positions. During the letter placement, one additional process of improvement is done. The averaging of letter positions reduces heavily bent labels. This creates a more connected label appearance and a smoother label course. The letters are averaged by setting the letter positions to the center of their predecessor and successor. After this step, the spaces between these letter positions do not comply with the associated glyph advances. To correct them, the positions are shifted towards their neighbor position accordingly. This averaging procedure is done for all letters with two neighbors. Since heavily bended labels are hardly readable and visually less appealing, a label is not rendered if the angle between its averaged letter positions exceeds a certain maximum.
The determined label and letter positions facilitate the execution of visibility tests in the next step.

5 VISIBILITY TESTS

In the previous step, labels and their letters are assigned to a certain position. If all of them were rendered, the result would be an overloaded scene with worse label readability and labels covering important information. To avoid this scenario, intersection tests are implemented to determine if a label is overlapping with others. Since the labels are rendered by descending importance and ascending distance to camera, only a less or equally important label could occlude another label. When a label overlaps with another, it is not rendered. If the importance values are equal, the distance to the camera plays the decisive role. For more efficiency, the intersection tests are split into three stages.

The first stage represents a search grid, which enables intersection tests only with labels in the same region. These regions are cells in a grid covering the entire domain. Only labels that are lying in the same cell as the current label are possible intersection candidates. The cell access is easy and fast and the search grid avoids a lot of unnecessary intersection tests.

The second and third stages are the intersection test itself with two different kinds of bounding boxes, which differ in their level of detail. After the first step, all possible intersection candidates need to be tested for overlap with the current label. At first, this is performed with axis-aligned label bounding boxes. These bounding boxes enclose all letters of a label and are only defined by four vertices. This makes them simple and fast, but inaccurate. The area covered by this kind of bounding box may be much larger than the label really needs, since it is always an axis-aligned rectangle.
By only using the label bounding boxes, there would be many false-positive intersections. Therefore, we use a second kind of bounding box for intersection testing, which is only performed if the first bounding boxes are intersecting. The individual letter bounding boxes are used to test for overlap of single letters. This is more precise, but needs more execution time because it has to be performed for each letter. If this last intersection test detects an intersection, the current label is not rendered.

The abrupt change of label status from visible to invisible and vice versa triggers label flickering. Therefore, a smooth fade-in or fade-out is performed by lowering or increasing label transparency on certain cases. It is possible to predict if a line label is running out of space because it may be located at the boundaries of its line segment. Then the transparency is gradually reduced according to the remaining space. In Figure 3, one can see the label "Bischofsgartenstraße" fading out, caused by the short space remaining of its corresponding line segment. As discussed earlier, labels that are far away are less interesting for the viewer than near labels. Due to this fact, we aim to make labels far away disappear.

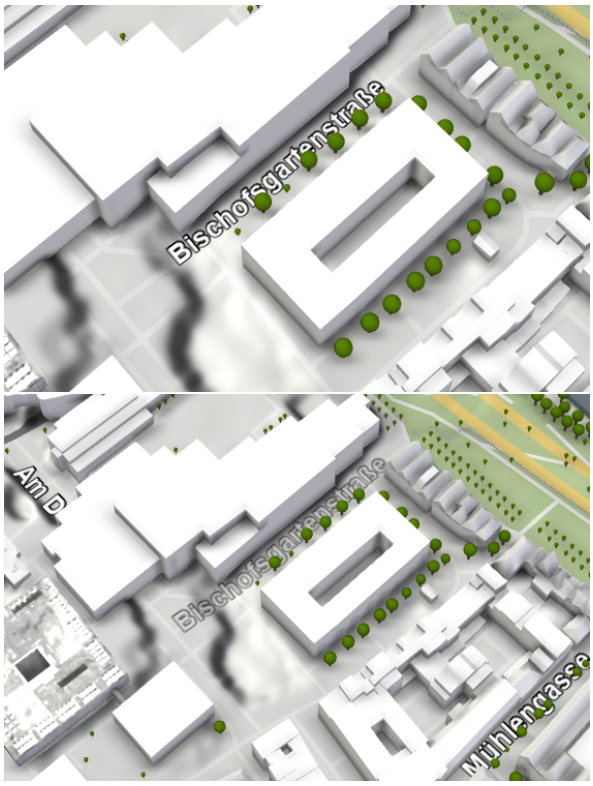


Figure 3: Fading out the label "Bischofsgartenstraße" because it is running out of space while zooming out.

This increases the possibility that labels that belong to streets, places and other landmarks near the viewer are visible. To also avoid flickering in this case, they are faded in and out like labels, which are running out of space.

Until now, we have only discussed intersections between labels, but a usual scene in Visdom includes many different objects like buildings, protection lines, trees and mountains that also influence the label visibility of line labels. Area labels are rendered floating above everything else, therefore we do not need to consider occlusion by other objects. It is difficult to deliver a realistic depth perception and make line labels visible through other objects at the same time. For perspective views, we decided to preserve the depth information for close line labels and give it up on growing distance. Labels have constant size with respect to the screen, but other objects have constant size with respect to the 3D world and they are getting very small if they are far away. Then their detailed visual information is no longer detectable. Therefore, it is less irritating that the labels are visible through occluding objects if they are far away. In the case of orthographic views, the depth of buildings and other objects, which may hide line labels, is hardly perceptible by the user. Thus, the annotations are always visible through them without destroying the 3D perception.
The visibility tests determine the labels passed to the

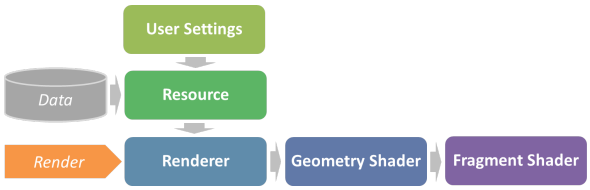


Figure 4: Overview of internal structure and information flow (gray arrows) of the render process.

last step of the annotation process to finally render them.

6 RENDERING

The last step of the annotation process is the rendering of all labels that passed the visibility tests. The orange arrow in Figure 4 visualizes the invocation of the *Renderer* class by the logic of Visdom to start a new render pass in each frame. When the *Renderer* becomes active, it starts the annotation render process with the data received by the *Resource* class, which manages the label data. These data have different sources and attributes. The data delivered by OpenStreetMap [Ope19] are constant at run-time and the glyph metrics extracted from a given font do not change either. Therefore these data represent the static part of the data managed by the *Resource*. The user settings represent the dynamic data because the values can be modified by the user during execution. If the user changes them, the *Resource* updates all dependent resources and provides the refreshed data for the *Renderer*. If there are no changes made by the user, the default font is the Google font Roboto [Rob19], which has a special design to make label text more legible even with a small point size.

To increase the contrast between labels and their background and to further improve their readability, outlines are added to the letters. The size and color information of a label indicate the label relevance for the user. The outline color is chosen as white or black, depending on which color gives the highest monochrome contrast to the font color. The default font is bold to balance the ratio between the label letters and their outline and to improve annotation text legibility.

According the current view, the *Renderer* uses the updated resource data for the label and letter placement and for the execution of visibility tests. After these steps, it passes label texts, letter positions, scale factors and the glyph metrics to the geometry shader to span the quad for every letter. This is done on the GPU to save important runtime. At this point, every character has only one position and the geometry shader uses the glyph metrics to calculate four vertices out of this one.

But since there is only one real point in the 3D world of Visdom, there is also only one depth value. For area labels this does not matter, since they do not need a precise depth value because they are not depth tested with

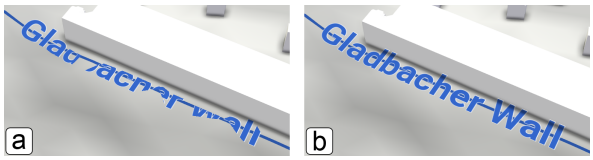


Figure 5: Occlusion issue with standard depth test (a) and desired behavior with modified depth test (b).

the rest of the scene. This is due to the fact that area labels are floating above everything else. But for line labels, the missing accuracy of depth values is a real problem. This one depth value per character lies directly on the line and if the underlying ground is not exactly parallel to the line, the four generated vertices out of this one appear wrong and are sometimes occluded by the terrain (see Figure 5a). To avoid such depth errors, the OpenGL integrated depth testing is disabled and an own depth test is performed in the fragment shader. The result of the own depth test implementation is shown in Figure 5b. This test uses a depth offset that complies with the letter height. If the letter depth with this offset is closer than the rest of the 3D environment, the letter will be drawn.

The *Renderer* passes color information and texture atlases of the characters and their outlines to the fragment shader. Afterwards, the fragment shader is able to finally draw the label characters to the global frame buffer of the Visdom scene.

7 RESULTS

The described annotation process of the last sections is implemented with the use of the graphics API OpenGL and the programming language C++. All used fonts by annotations in Visdom are of the font format TrueType. It is a common font format that stores font information by the use of quadratic Bézier splines. This classifies it to the group of vector graphic font formats. To access and handle the font data provided by TrueType, the library FreeType [Lem19] is used. It eases the extraction of glyph metrics and the rasterization of vector graphics to get appropriate bitmap font data. If the used font does not provide hinting information, FreeType offers autohinting to make text more legible. But missing kerning information cannot be replaced and is therefore a strict requirement for the chosen font.

We use different case studies to analyze the visual appearance and behavior of the annotations in Visdom. There are five flood management case studies, three of them are located in Austria, one in Germany and one in Italy. They differ in attributes such as landscape, amount and type of label data and in the case of Italy, in language of annotations. The case studies do not only consist of densely populated cities but some also cover a wider range with multiple cities and villages. Some also have hills, mountains, rivers and dikes, where the

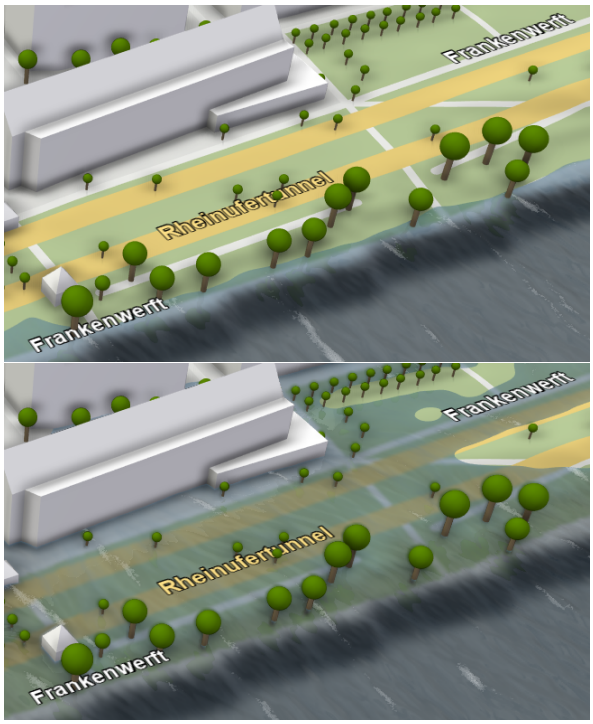


Figure 6: Line labels dynamically adapt to the water surface of the flood simulation to stay afloat.

annotations are applied like in cities. These varied case studies offer the opportunity to see how the annotations act when the environment and the provided label data are changing. They ease the analysis of the overall strengths and weaknesses of the annotations. For this purpose, performance tests are executed in the five case studies in different scenarios. These scenarios cover different views, from orthographic to perspective side views and from high to low zoom levels.

7.1 Case studies

The fist case study is Cologne in which the annotations where initially implemented. There is a high amount of label data to process, which is a challenge for their dynamic real-time rendering. The terrain is relatively even with only very low height. This fact improves the appearance of line labels, because their individual letters are not changing orientation heavily when adapting to the underlying ground. In the Cologne case study, the annotations are not only used for streets, buildings, places and landmarks, they are also applied to labeling protection barriers and sewer networks. In Figure 6 a small part of Cologne is shown. One can see labels rising with the flood, which enables their legibility during the entire flood simulation, even in inundated areas.

The case study of HORA (= Natural Hazard Overview & Risk Assessment Austria [Bun18]) sticks out with the most mountains and rivers in the valleys. The area of Tyrol shown in Figure 7a is characterized by the mountains of the Alps. The rivers are emphasized with light blue lines for better river visibility during flooding. One can see that the floating area labels are well-suited for city and village labels in mountainous landscapes. But when labeling rivers and streets on mountain slopes or sharp ridges, the labels may be heavily bended and thus hardly readable or disappear when reaching the maximum bend angle.

The use of the annotations in narrow valleys or on hills can be observed in the case study of Wachau (see Figure 7b). In addition, the Figure shows the annotation of the large Danube river along its center line as well as the combination of labeling flat areas in the valley and uneven ground on the hills. Compared to the HORA case study, the labels are disappearing less often since the hills of Wachau do not have such steep slopes as the mountains of Tyrol.

The dynamic adaption of the label orientation to make text always readable from left to right is still correct on borderline cases such as the almost vertical labels in Figure 7c. The labels are still easy to read even if they nestle up to gentle hills and bend along curvy streets. The Marchfeld case study uses annotations also to label dikes.

The last of the five introduced case studies is Florence. The annotations behave similarly to Cologne because they are both cities with relatively even ground. Florence has narrower streets and is populated more densely than Cologne. Thus, there are more streets over the same area and therefore more labels to process. Figure 7d shows the labels of Florence in orthographic view where the modified depth test shows its positive effect. Using the standard depth test would result in partially hidden street labels behind buildings. One can also see that the color-coded buildings and terrain attract attention but the labels are still clearly visible and legible.

An effect, which is visible in all five case studies is that the annotations are changing their level of detail according to the zoom factor. This property is a positive side effect of the implemented visibility tests and consideration of label importance. When zooming out, labels which are less important are disappearing and give way to more important labels. By zooming out, more labels are overlapping with each other and only the labels with higher relevance have the privilege to be rendered. When zooming in, fewer labels are overlapping and less important labels get enough space and become visible.

7.2 Quality

In the related work section the four quality criteria aesthetics, label visibility, feature visibility and label-feature association for label placement of Van Dijk [DKSW02] were introduced. In the following sections the implemented annotations are analyzed according to them.

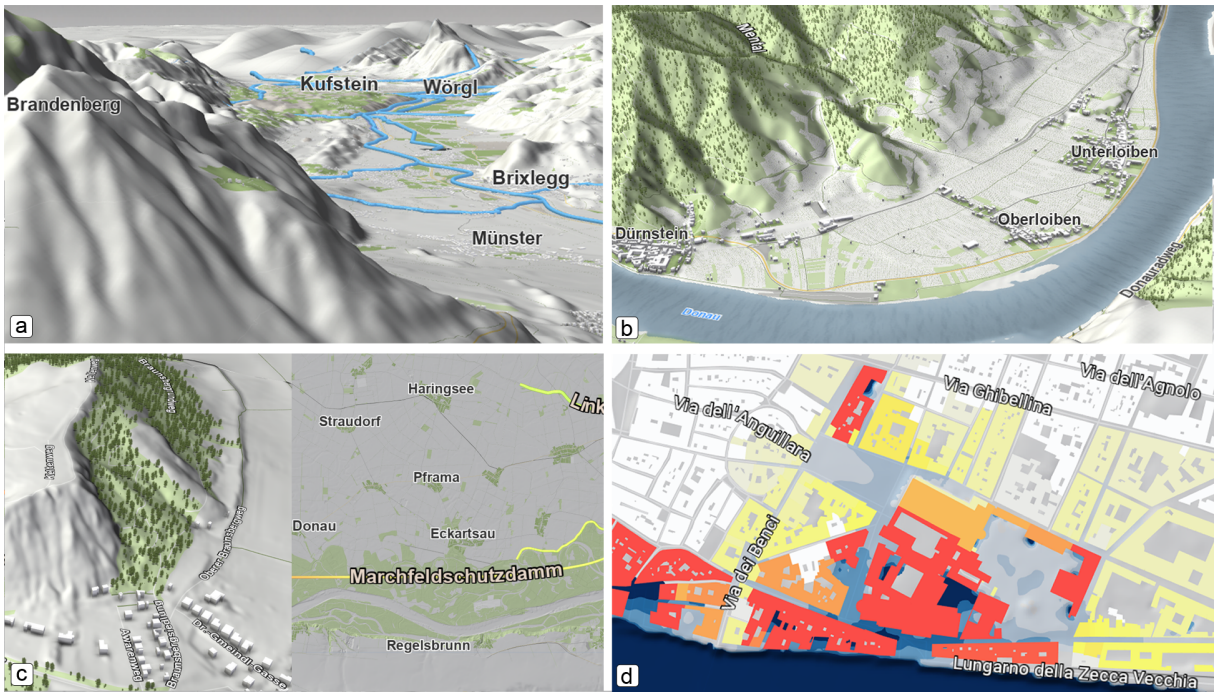


Figure 7: The first case study (a) shows the annotations used in the mountainous environment of Tyrol. In Wachau (b) one can see labels in narrow valleys and used for the Danube river. The Marchfeld case study (c) shows labels nestling up to hills and used for dikes. The last case study (d) depicts an orthographic view of Florence with Italian labels.

Aesthetics

The chosen font "Roboto" provides kerning and hinting information to get a clean and sharp text that is easily legible even on small point sizes. Van Dijk stated that the label should follow the reading direction and should be as horizontal and straight as possible. For area labels, this is obviously true since they are always drawn horizontally from left to right. Due to dynamically adapting the label orientation and due to the averaging algorithm on letter positions, the line labels comply with this criterion too.

Label & Feature Visibility

Since the labels do not appear if they are overlapped by other labels they, are only covered by other objects of the environment. In this case we made a compromise between the preservation of depth perception and label visibility by occluding labels in detailed views only. The label visibility criterion of Van Dijk that labels should be placed either on land or on water is realized by the concept of floating labels. Either the label lies on the ground or it is floating on the water.
Fix positions of all labels have a positive effect on performance but they lead to truncation of label text and invisible labels on long streets. The visibility of label features should be enhanced by avoiding a label placement by which important features are covered, overlapped, concealed or disturbed by labels. The relevant information of streets is their location, which is not hidden by labels projected on them. Area labels cover scene parts that may include important information but floating labels enhance high readability and fast identification of their belonging features. During the use of Visdom, area labels are not perceived as obstructive.

Label-Feature Association

Line and area labels are easy to differentiate by their visual appearance, because line labels are projected onto their feature and area labels are always floating above them. Thus, a mix-up of line and area labels is unlikely. The direct embedding of line labels into their features creates a very high label-feature association. The labeling of line segments before and after crossings and junctions further improves the matching of labels to their belonging sections. According to Van Dijk, area labels should have an extent that corresponds to the shape of its feature. The diverse shapes and sizes of areas provided by OpenStreetMap [Ope19] make an adaption complicated and labels are sized according to their importance.

7.3 Performance Tests

In this section, the results of performance tests on the introduced case studies are presented. The tests were executed on a computer with 32 GB RAM and Intel Core i5-4690K CPU with 4x3.50 GHz and with a Nvidia Geforce GTX 980 graphic card with 4 GB.

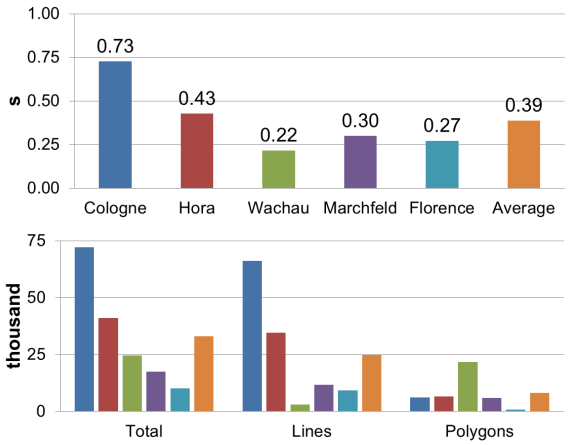


Figure 8: The time needed to update all label data (top) and the distribution of shapes (bottom), both per case study.

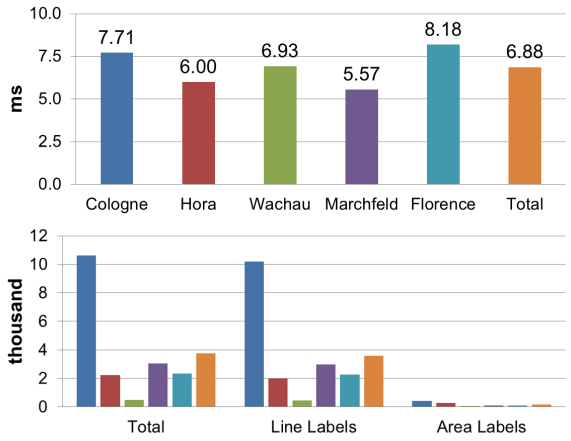


Figure 9: The average time needed for one render pass per case study (top). The number of line and area labels and the total number of labels per case study (bottom).

The tests are separated into the two categories update and render duration. The update process is executed only on the CPU. By counting CPU cycles, the elapsed time is measured. For the update tests, only the initial update process involving the entire data set is considered because later updates are only processing subsets of the data. The render duration takes into account the time needed for the placement, the visibility tests and the rendering itself. These processes are performed on both CPU and GPU. The elapsed GPU time is measured with a pipeline statistics query exposed by the OpenGL API.

The distribution of shapes over all case studies is visualized in Figure 8 (bottom). Wachau is the only case study that has more polygons than lines. In all other case studies, line data dominate, which requires more line merges during the initial update process. The number of lines and the update time are strongly correlated. For the case study of Cologne, OpenStreetMap provides the most label and shape data of all case studies. In Figure 8 one can see, that the initial preprocessing of all label data of the largest case study with over 70,000 shapes (bottom) is faster than one second (top).

The average time needed to render these labels is visualized in Figure 9 (top). The number of labels rendered to the screen lies between 20 and 30 labels on average. Since the most labels are rendered in the case study of Florence and the least in Marchfeld, they have the longest and shortest average render duration. In Figure 9 (bottom) the number of labels after the update process is shown. Compared with Figure 8 (bottom) there are far less labels than shapes because labels with invalid text are discarded and line labels merged. Focusing on the Cologne case study, the processing of over 10,000 labels and the rendering of labels passing the visibility tests took 7.71 ms.

These results show that all case studies can be rendered at highly interactive frame rates and that the annotations allow fluent dynamic real-time interaction with Visdom.

8 SUMMARY & FUTURE WORK

This paper covers the implementation of annotations for an interactive 3D application. Two different label approaches for line and area labels facilitate an object-specific labeling, which enables a direct label-feature association without additional objects. The dynamic adaption of the label orientation to user interaction and changing views increases the label legibility. Through the implemented occlusion handling, the depth perception in close-up views is preserved and in far-away views the labels are still readable. The annotations provide continuous levels of detail, whereby the label visibility depends on the label importance and the zoom level. The scaling according to the zoom level fixates the label size relative to the screen and preserves label legibility.

The implementation supports processing large quantities of labels in real time and with a fluent user interaction. The use of fixed label positions speeds up the placement process because only the individual letters need to be placed dynamically and not the whole label. The preparation of all data that are not changing permanently is also saving important runtime. The three step visibility tests with the search grid and label and letter bounding boxes represent a further performance improvement. Finally, the GPU-side vertex calculation of the label letters enables the use of only one position per letter for CPU processing, which also contributes to the high interactivity. Thus, the implementation fulfills all measurable initial requirements.

The different case studies demonstrate that the annotations are applicable to different scenarios. But there are some known issues. The line merge process does not reliably detect all crossings and junctions to provide an optimal label-feature association.

The problem of one depth value per letter causes imprecise depth tests with the environment. This can lead to labels partially sinking into the ground or labels visible in front of objects, which should cover them. This behavior is only appearing in very special cases such as a highly uneven terrain. As a result of the fixed label positions, the labels are often truncated or not visible even if their feature is. This is noticeable especially in the case of long streets with only one label.

We consider a more flexible, but temporally stable label placement for very long streets an important direction for future work.

9 ACKNOWLEDGMENTS

This work was enabled by the Competence Centre VRVis. VRVis is funded by BMVIT, BMWFW, Styria, SFG and Vienna Business Agency in the scope of COMET – Competence Centers for Excellent Technologies (854174) which is managed by FFG.

10 REFERENCES

[Bun18] Bundesministerium für Nachhaltigkeit und Turismus. Natural Hazard Overview & Risk Assessment Austria. http://www.hora.gv.at, 2019. Accessed: 2019-01-15.

[DKSW02] Steven Van Dijk, Marc Van Kreveld, Tycho Strijk, and Alexander Wolff. Towards an evaluation of quality for names placement methods. *International Journal of Geographical Information Science*, 16(7):641–661, 2002.

[Geo18] Geofabrik GmbH. Maps & Data. http://www.geofabrik.de/data, 2018. Accessed: 2019-01-15.

[Gre07] Chris Green. Improved Alpha-tested Magnification for Vector Textures and Special Effects. In *ACM SIGGRAPH 2007 courses*, pages 9–18, 2007.

[Gus12] Stefan Gustavson. 2D Shape Rendering by Distance Fields. In *OpenGL Insights: OpenGL, OpenGL ES, and WebGL community experiences*, pages 173–182. CRC Press, 2012.

[Lem19] Werner Lemberg. FreeType Overview. https://www.freetype.org/freetype2/docs, 2019. Accessed: 2019-01-15.

[MD07] Stefan Maass and Jürgen Döllner. Embedded Labels for Line Features in Interactive 3D Virtual Environments. In *Proceedings of the 5th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, pages 53–59, 2007.

[MD08] Stefan Maass and Jürgen Döllner. Seamless Integration of Labels into Interactive Virtual 3D Environments Using Parameterized Hulls. In *Proceedings of the Fourth Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging*, pages 33–40, 2008.

[MS91] Joe Marks and Stuart Shieber. The Computational Complexity of Cartographic Label Placement. Technical report, Harvard Computer Science Group, 1991.

[NN16] Benjamin Niedermann and Martin Nöllenburg. An algorithmic framework for labeling road maps. In *The Annual International Conference on Geographic Information Science*, pages 308–322. Springer, 2016.

[Ope19] OpenStreetMap Contributors. OpenStreetMap. https://www.openstreetmap.org, 2019. Accessed: 2019-01-15.

[Rob19] Christian Robertson. Google Fonts Roboto. https://fonts.google.com/specimen/Roboto, 2019. Accessed: 2019-01-15.

[Rou13] Nicolas P. Rougier. Higher Quality 2D Text Rendering. *Journal of Computer Graphics Techniques*, 2(1):50–64, 2013.

[SLL+17] Jiangfeng She, Jianlong Liu, Chuang Li, Jiaqi Li, and Qiujun Wei. A line-feature label placement algorithm for interactive 3d map. *Computers & Graphics*, 67:86–94, 2017.

[SWH14] Nadine Schwartges, Alexander Wolff, and Jan-Henrik Haunert. Labeling streets in interactive maps using embedded labels. In *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 517–520. ACM, 2014.

[TKGS14] Markus Tatzgern, Denis Kalkofen, Raphael Grasset, and Dieter Schmalstieg. Hedgehog labeling: View management techniques for external labels in 3D space. In *2014 IEEE Virtual Reality (VR)*, pages 27–32, 2014.

[VFW13] Mikael Vaaraniemi, Martin Freidank, and Rüdiger Westermann. Enhancing the Visibility of Labels in 3D Navigation Maps. In *Progress and New Trends in 3D Geoinformation Sciences*, pages 23–40. Springer Berlin Heidelberg, 2013.

[VTW12] Mikael Vaaraniemi, Marc Treib, and Rüdiger Westermann. Temporally Coherent Real-time Labeling of Dynamic Scenes. In *Proceedings of the 3rd International Conference on Computing for Geospatial Research and Applications*, pages 17:1–17:10, 2012.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Identification of stereo rig alignment error based on vertical disparity map

Sandeep Katragadda
KiwiSecurity Software GmbH,
Guglgasse 15, 1110, Vienna, Austria
s.katragadda@kiwisecurity.com

ABSTRACT

Poor quality of 3D video content can lead to headache, blurry vision and overall exhaustive experience for the viewer. To ensure quality and comfortable 3D experience for the end consumer, common production errors must be detected and corrected. Vertical disparity is one of these distortions and is caused by improper stereo-camera setup. This paper aims at identifying the possible rotational and placement errors that cause the vertical disparity. An estimation of these errors is necessary to produce good quality 3D content. According to my knowledge, there exists no method to identify rig alignment errors without the knowledge of camera setup parameters and this work is the first step in that direction. Feature point detection has proven to be an interesting approach to find vertical disparity present in the given stereo image pair. In this work feature extraction techniques such as SIFT, SURF and Harris features are efficiently used to compute reliable and robust vertical disparity patterns. This paper classifies vertical disparity patterns according to rig errors. If the vertical disparity values are too small or ambiguous to be identified by pattern analysis, this paper uses graphical analysis that highlights the relationship between the total vertical disparity and the contribution of each possible error to the total. Experimental results show that the proposed approach identifies the reason behind the presence of vertical disparity of a stereo image pair.

Keywords

Stereoscopic 3D, feature matching, vertical disparity, epipolar constraint, RANSAC.

1 INTRODUCTION

Experiencing 3D is one of the most important perception features of human kind. Several depth cues are used by human visual system to perceive depth and to experience 3D. Stereoscopic video systems simulate the human binocular vision [Sammons92]. Binocular depth cues are observed by both eyes. They include retinal disparity in the two views of a scene (one from each eye) resulted from inter-ocular distance, and convergence due to the inward movement of both eyes to focus at a point of interest. If the cues are not reproduced properly in the stereo content, it results in a bad user experience [Perek16, Tam11, Hodges91]. Poor quality of stereoscopic video content results in visual fatigue [IJsselsteijn00, ATSC11]. Consequences of visual fatigue on human 3D perception might be eye strain, watery eyes, nausea, head aches, focusing difficulty and blurred vision. Therefore, stereographers and camera teams have to take into account a variety of conditions, guidelines and rules right from the beginning of the production chain to create good stereoscopic video content [Zilly11, Knorr12]. It includes accurate rigging and calibration of the stereo cameras, good adjustment and matching of electronic and optical camera parameters as well as the adaption of the stereo baseline to the depth structure of the scene content.

When there is an error in the alignment of the stereo rig setup, vertical disparity can be observed in the image pair. Vertical disparity is defined as the vertical displacement between corresponding pixels in the left and right images. Vertical disparity in the stereo content is result of distortions. There are other distortions such as keystone distortion, depth plane curvature, barrel distortion and pin cushion distortion caused by other factors. One of the results of vertical disparity is vertical parallax which can cause eye strain, visual fatigue, confusion and loss of stereopsis. The alignment errors in the stereo rig setup include rotational errors, translational errors or both. The rotational errors include relative roll, relative pan and relative tilt between the cameras. The translational errors include relative vertical shift and relative forward or backward shift between the cameras. One more error is zoom difference between the two cameras. It is caused when the lenses used in the two cameras have different focal length. Converged

(toed-in) camera setup and parallel camera setup are the two stereoscopic camera configurations used in 3D capturing [Woods93]. In the toed-in setup, the two cameras are angled towards each other resulting in a finite convergence distance (the distance at which the two camera images coincide in the centre of the stereoscopic display). The setup causes keystone distortion and vertical disparities which can be corrected using stereo image rectification [Faugeras93] during post processing. In the parallel setup, the optical axes of the two cameras are parallel so the convergence distance is infinity. Horizontal Image Translation (HIT) is done during post processing to place the point of convergence wherever desired. In the ideal case, the parallel setup does not produce vertical disparity.

This paper mainly focuses on vertical disparity caused by stereo rig alignment errors in the parallel setup and proposes a method to identify rig alignment errors based on vertical disparity in the given stereo image pair. These errors include relative rotation, relative translation and relative zoom difference. It is assumed that no lens distortion is present in the given image pair and the image pair is not rectified after capturing. The steps include reliable estimation of the vertical disparity and finding what kind of stereo rig error is causing this vertical disparity. The main contributions are (1) classification of possible vertical disparity patterns according to the stereo rig errors, and (2) identification of rig alignment error using vertical disparity map.

The paper is organised as follows. Section 2 briefly presents the state of the art on stereo rig alignment error identification. Section 3 presents the required mathematical background of stereo capturing. Section 4 proposes the vertical disparity based alignment error identification method. Section 5 presents the experimental results. Finally, Section 6 concludes the paper.

2 STATE OF THE ART

For a calibrated system, there is a lot of work presented in the literature to identify rig alignment errors. Behrooz Kamgar-Parsi et al. [Kamgar-Parsi88] computed small rotational errors caused by mechanical difficulties in the stereo rig setup. They proved that an error of 0.5 degrees in pan angle can result in an error of 4 meters in depth estimation which emphasizes the importance of accurate alignment. They assumed that both cameras are calibrated, i.e., the focal length, spatial digitisation of image plane and lens distortion are known. Behrooz's work also emphasizes that the points selected from a specific region cannot cover all properties of relative rotations. Hence, it is important to select point correspondences from different regions of the image. In case of uncalibrated camera systems, identification of stereo rig error is still a challenge. Richard I. Hartley [Hartley92] and Frederik Zilly et al. [Zilly10] proposed methods to compute the alignment errors from an uncalibrated stereo by using some assumptions on intrinsic parameters. Hartley did a lot of research in multiple view geometry [Hartley04]. He proposed two algorithms to calculate the relative placement of the cameras in the stereo rig [Hartley92]. One algorithm deals with the calibrated camera system and the other deals with the uncalibrated camera system. In case of calibrated system, it is assumed that everything is known except relative positions and relative orientations. In case of uncalibrated system, it is assumed that the two cameras are of the same type (having the same intrinsic parameters and the same lenses) and the principle point is at the image centre, the focal lengths of the two cameras, their relative placement and orientation are determined. This approach results in imaginary values when the assumptions fail to satisfy. Hence the result is not guaranteed in all cases. Frederik Zilly et al. proposed a new method [Zilly10] that computes rectifying homographies using sparse stereo correspondences from an uncalibrated stereo. This paper uses this approach to classify the disparity patterns caused by different alignment errors and to identify the stereo rig errors.

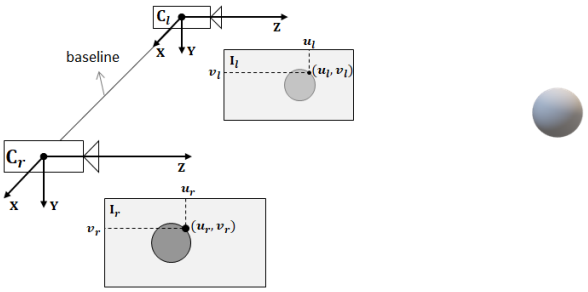


Figure 1: Stereo camera geometry in parallel setup. Key - $\mathbf{C}_l$ and $\mathbf{C}_r$ are optical centres of left and right cameras respectively. $\mathbf{I}_l$ and $\mathbf{I}_r$ are the left and right image planes respectively. (u_l, v_l) and (u_r, v_r) are the pixel locations of a real point on left and right image planes respectively.

3 MATHEMATICAL BACKGROUND

Consider a parallel stereo setup shown in Figure 1 and 2. The line joining the optical centres $\mathbf{C}_l$ and $\mathbf{C}_r$ of the left and right cameras respectively is called baseline. Consider a point $\mathbf{P}$ in space whose perspective projection is at point $\mathbf{p}_l = (u_l, v_l)$ on the image plane $\mathbf{I}_l$ of left camera and at point $\mathbf{p}_r = (u_r, v_r)$ on the image plane $\mathbf{I}_r$ of right camera. The plane connecting $\mathbf{P}$, $\mathbf{C}_l$ and $\mathbf{C}_r$ is called epipolar plane. The plane intersects the image planes $\mathbf{I}_l$ and $\mathbf{I}_r$ in lines called epipolar lines. Let $\mathbf{P}_r = [X_r\ Y_r\ Z_r]$ represent the same point $\mathbf{P}$ in the right camera coordinate system. The perspective projection of the point $\mathbf{P}_r$ on the image plane $\mathbf{I}_r$ is at $(f_r X_r/Z_r, f_r Y_r/Z_r)$ where f_r is the focal length of the right camera. To obtain this image location in terms of pixel values, intrinsic parameters of the camera have to

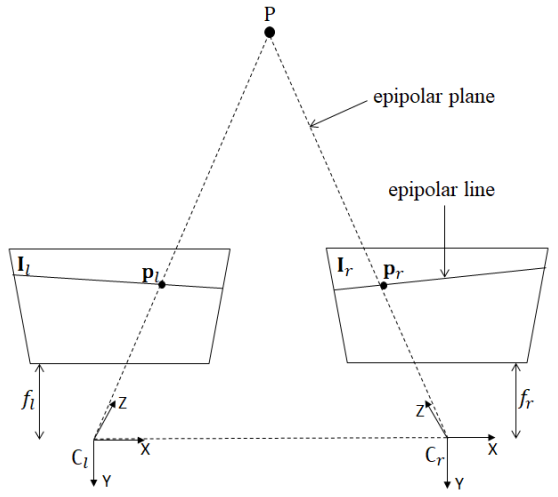


Figure 2: Epipolar geometry. Key - $\mathbf{C}_l$ and $\mathbf{C}_r$ are optical centres of left and right cameras respectively. $\mathbf{I}_l$ and $\mathbf{I}_r$ are the left and right image planes respectively. $\mathbf{P}$ is a point in the scene. $\mathbf{p}_l$ and $\mathbf{p}_r$ are the pixel locations of $\mathbf{P}$ on left and right image planes respectively. f_l and f_r are the focal lengths of the left and right cameras respectively.

be considered. They include principle point [Clarke98], pixel dimensions [Lenz88] and skew coefficient describing non-rectangular pixels. If $(c_r^x,\ c_r^y)$ is the pixel coordinate of the principle point, s_r^x and s_r^y represent effective pixel sizes along the horizontal and vertical directions of the image respectively [Lenz88], the 3D point $\mathbf{P}_r$ to 2D point $\mathbf{p}_r$ transformation is [Hartley04]

$$\begin{aligned} u_r &= (f_r X_r/Z_r)/s_r^x + c_r^x, \\ v_r &= (f_r Y_r/Z_r)/s_r^y + c_r^y. \end{aligned} \tag{1}$$

An upper triangular matrix describing these intrinsic properties of the right camera holding the parameters such as principle point (c_r^x, c_r^y), pixel dimensions f_r/s_r^x and f_r/s_r^y, and skew s_r is defined as

$$\mathbf{K}_r = \begin{bmatrix} f_r/s_r^x & s_r & c_r^x \\ 0 & f_r/s_r^y & c_r^y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

Assuming zero skew, equation 1 can be written as

$$\begin{bmatrix} \mathbf{p}_r \\ 1 \end{bmatrix} = \mathbf{K}_r \hat{\mathbf{P}}_r, \tag{3}$$

where $\hat{\mathbf{P}}_r = [X_r/Z_r\ Y_r/Z_r\ 1]^\intercal$ represents the projection on the right image plane (the superscript $\intercal$ refers to the transpose operation). Similarly, $\mathbf{p}_l$ is image point (u_l, v_l) of the same world point $\mathbf{P}$ whose 3D location is $\mathbf{P}_l$ in the left camera coordinate system and $\hat{\mathbf{P}}_l$ represents the projection on the left image plane.

Epipolar geometry describes the geometric relationship between a pair of stereo images [Boufama95]. Let the left camera coordinate system be the reference coordinate system, $\mathbf{R}$ be the 3×3 matrix (the rotation matrix) defining the relative rotation of the right camera with respect to the left camera and $\mathbf{T}$ be the 3×1 vector (the translation vector) defining the relative translation of the right camera with respect to the left camera. Then the relation between $\hat{\mathbf{P}}_r$ and $\hat{\mathbf{P}}_l$ is $\hat{\mathbf{P}}_r = \mathbf{R}(\hat{\mathbf{P}}_l - \mathbf{T})$[Andrea00]. From the plane equation of two vectors, the equation of epipolar plane is

$$\begin{aligned} (\hat{\mathbf{P}}_l - \mathbf{T})^\intercal.(\mathbf{T} \times \hat{\mathbf{P}}_l) = 0, \\ (\mathbf{R}^\intercal \hat{\mathbf{P}}_r)^\intercal [\mathbf{T}]_\times \hat{\mathbf{P}}_l = 0. \end{aligned} \tag{4}$$

Here $[\mathbf{T}]_\times$ is the skew-symmetric matrix of vector $\mathbf{T}$.

From equation 3 and equation 4,

$$\begin{bmatrix} \mathbf{p}_r & 1 \end{bmatrix} \mathbf{F} \begin{bmatrix} \mathbf{p}_l \\ 1 \end{bmatrix} = 0, \tag{5}$$

where $\mathbf{F} = (\mathbf{K}_r^{-1})^\intercal \mathbf{R} [\mathbf{T}]_\times \mathbf{K}_l^{-1}$ is called fundamental matrix between the pair of stereo images. Equation 5 is called Epipolar constraint. This maps points from one image to corresponding epipolar lines in the other image. The fundamental matrix $\mathbf{F}$ encodes the information of both extrinsic ($\mathbf{R}$ and $\mathbf{T}$) and intrinsic parameters ($\mathbf{K}_l$ and $\mathbf{K}_r$). A stereo camera system is said to be calibrated if its intrinsic parameters and the extrinsic parameters are known. If some of the parameters are known it is called semi calibrated or partially calibrated. If nothing about these parameters is known the system is an uncalibrated system.

In the ideal parallel stereo setup (parallel Z-axis of the cameras) case, the X-axis of each camera is colinear with the base line and hence projects the point $\mathbf{P}$ on to the two image points $\mathbf{p}_r$ and $\mathbf{p}_l$ on the same scan line making the vertical disparity $v_r - v_l$ zero. The distance between the point $\mathbf{P}$ and the stereo camera is proportional to the horizontal disparity $u_r - u_l$.

4 PROPOSED METHOD

Let α_x, α_y and α_z are the relative rotation angles of the right camera with respect to the left camera around X, Y and Z axes respectively, and t_x, t_y and t_z are the relative translation of right camera with respect to the left camera along X, Y and Z axes respectively. This work assumes that (1) the relative rotation angles are small, i.e. $\alpha_x << 1$, $\alpha_y << 1$ and $\alpha_z << 1$, and the rotation matrix $\mathbf{R}$ can be approximated as

$$\hat{\mathbf{R}} = \begin{bmatrix} 1 & -\alpha_z & \alpha_y \\ \alpha_z & 1 & -\alpha_x \\ -\alpha_y & \alpha_x & 1 \end{bmatrix}, \tag{6}$$

(2) relative translation in Y and Z directions are small compared to that in X-direction (baseline length), i.e. $\hat{t}_y = t_y/t_x << 1$ and $\hat{t}_z = t_z/t_x << 1$, so without affecting Equation 5, $\mathbf{T}$ can be taken as $\hat{\mathbf{t}} = [1\ \hat{t}_y\ \hat{t}_z]^\intercal$ and

$$[\hat{\mathbf{t}}]_\times = \begin{bmatrix} 0 & -\hat{t}_z & \hat{t}_y \\ \hat{t}_z & 0 & -1 \\ -\hat{t}_y & 1 & 0 \end{bmatrix}, \tag{7}$$

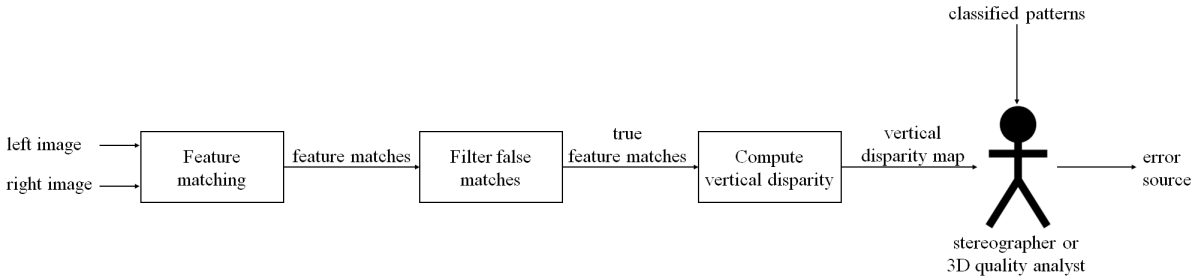


Figure 3: Block diagram showing the steps in the proposed classification based rig alignment error identification approach.

(3) pixels are square shaped meaning the skew of the cameras is zero (i.e. $s_r = s_l = 0$), $s_r^x = s_r^y$ and $s_l^x = s_l^y$, (4) focal lengths' difference α_f is very small such that $\hat{f}_r = (1-\alpha_f)/\hat{f}_l$ where $\hat{f}_l = f_l/s_l^x$ and $\hat{f}_r = f_r/s_r^x$ are the focal lengths of left and right cameras (in pixels) respectively.

By substituting the assumptions in Epipolar constraint (equation 5) and ignoring the second and higher order terms, the total vertical disparity $v_r - v_l$ corresponding to the point **P** can be written as a function of $(\hat{f}_l, \alpha_f, \alpha_x, \alpha_y, \alpha_z, \hat{t}_y, \hat{t}_z)$ as [Zilly10]

$$v_r - v_l = \underbrace{\hat{t}_y(u_r - u_l)}_{Y-shift} + \underbrace{\frac{\hat{t}_z}{\hat{f}_l}(u_l v_r - u_r v_l)}_{Z-shift} + \underbrace{\alpha_f v_r}_{zoom\ difference} \underbrace{-\alpha_x \hat{f}_l - \frac{\alpha_x}{\hat{f}_l} v_l v_r}_{tilt} + \underbrace{\frac{\alpha_y}{\hat{f}_l} u_r v_l}_{pan} + \underbrace{\alpha_z u_r}_{roll}. \tag{8}$$

It can be seen from the equation that different errors produce different vertical disparity patterns in the image. The following section shows the disparity patterns caused by some errors. If stereographers or quality analysts are familiar with these patterns, they can identify the error causing the pattern in a given stereo image pair. Vertical disparity map can be obtained by using sparse features matching algorithms such as SIFT [Lowe04], SURF [Bay08] and Harris features. Figure 3 shows the block diagram of the proposed pattern classification based error identification approach.

4.1 Classification of disparity patterns

It has been shown that without the knowledge of the exact focal length of each camera in the stereo rig, computation of exact relative angles and relative positions is not possible. Moreover the focal length changes with changes in zoom. However, irrespective of the focal length value used to capture a scene, properties of the vertical disparity map are unique for each kind of error. Using equation 8, different stereo rig errors are classified according to the properties of the vertical disparity map. These properties can be used to identify the error sources based on the vertical disparity pattern present in the given stereo image pair. This section discusses the vertical disparity map properties for all possible stereo rig errors.

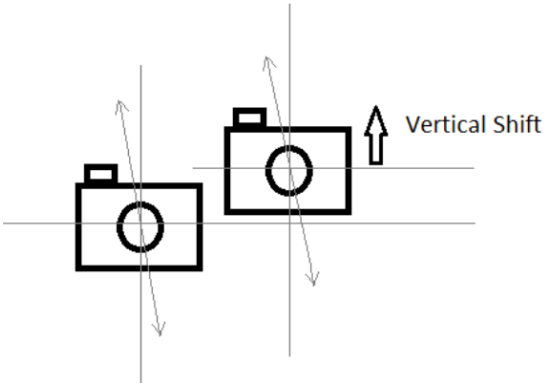


Figure 4: Cameras are at different heights

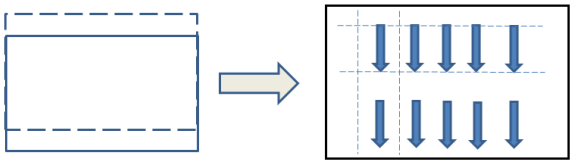
4.1.1 *Relative translation in Y-direction*

If the two cameras in a stereo setup are placed at different heights (X-axes of the cameras are separated vertically as shown in Figure 4), one image is shifted upward or downward relative to the other depending on the vertical shift direction. If the right camera is placed above the left camera, all left image features are moved downwards in the right image. See Figure 5(a) where the dashed border shows the right image and the thick border shows the left image. The vertical disparity is represented using arrows. Each arrow's head and tail locate at the row value (y- coordinates) of corresponding feature pixel in the right and the left images respectively. Similarly when the right camera is placed below the left camera, all left image features are moved upwards in the right image as shown in the Figure 5(b).

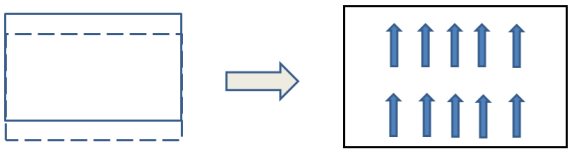
If vertical translation is the only error present in a stereo rig, equation 8 can be reduced to

$$v_r - v_l = \hat{t}_y(u_r - u_l), \tag{9}$$

where $u_r - u_l$ is horizontal disparity representing the depth. The properties of this error are: (1) vertical disparity is directly proportional to the horizontal disparity (depth) magnitude and (2) the value of vertical disparity is the same for all points located at a particular depth. These vertical disparity patterns can be seen in Figure 6.

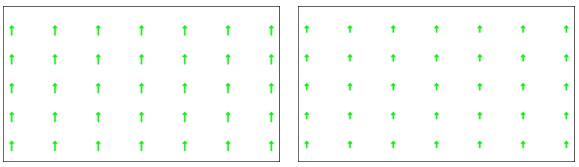


(a) Negative shift



(b) Positive shift

Figure 5: Placement error along Y-axis of left camera (thick rectangle) and right camera (dashed rectangle), and the corresponding vertical disparity pattern. The distance between dashed lines in pattern represents the magnitude of disparity.



(a) Depth = -600 (b) Depth = -400

Figure 6: Vertical disparity patterns at different depths ($u_r - u_l$) when the cameras are at different heights ($\hat{t}_y = 0.1$).

4.1.2 *Relative translation in Z-direction*

If one camera is placed a little forward or backward (along Z-axis) to the other camera, it results in vertical disparity. Figure 7 helps to visualise this type of setup error. If focal lengths of the two cameras are the same, a disparity pattern as shown in Figure 8(a) and Figure 8(b) can be observed. If translation in Z-direction is the only error present in a stereo rig, equation 8 can be reduced to

$$v_r - v_l = \frac{\hat{t}_z}{\hat{f}_l}(u_l v_r - u_r v_l). \tag{10}$$

The properties of this error are: (1) as the vertical disparity depends on u_r and u_l values, the disparity map varies with the depth, (2) the value of vertical parallax increases or decreases from the image top edge to the bottom edge, (3) the vertical disparity decreases as the depth increases and (4) vertical parallax is symmetrical about the row at the image centre. These vertical disparity patterns can be seen in Figure 9. Disparity due

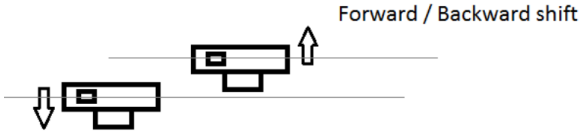
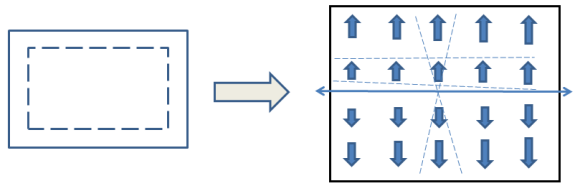
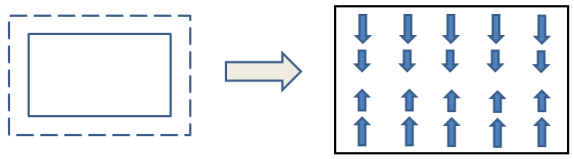


Figure 7: Cameras with a shift along the optical axis

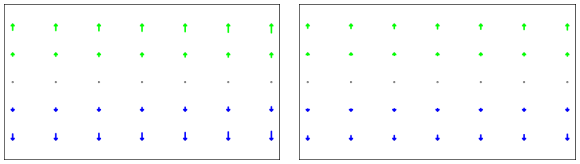


(a) Positive shift



(b) Negative shift

Figure 8: Placement error along Z-axis of left camera (thick rectangle) and right camera (dashed rectangle), and the corresponding vertical disparity pattern. The distance between dashed lines in pattern represents the magnitude of disparity.



(a) Depth=-600 (b) Depth=-400

Figure 9: Vertical disparity patterns at different depths ($u_r - u_l$) when the cameras are separated in Z-direction ($\hat{t}_z = 0.1$).

to translation in Z-direction decreases with increasing depth and becomes zero at infinity (i.e. at depth=0).

4.1.3 *Zoom difference*

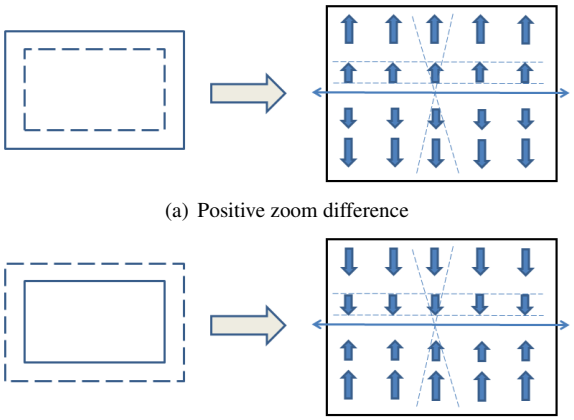
Another possible alignment error is focal length error. If two cameras use slightly different focal length values (different zoom), it shows similar vertical disparity pattern as previous (shift in Z-direction). See Figure 10(a) and Figure 10(b) to visualise the effect. If relative zoom is the only error present in a stereo rig, equation 8 can be reduced to

$$v_r - v_l = \alpha_f v_r. \tag{11}$$

The properties of relative zoom error are: (1) the disparity map is independent of depth information. This is the key point that differentiates this error from translational error in Z- direction where the vertical parallax varies with depth, (2) the amount of vertical parallax increases or decreases from the image top to the bottom and (3) The magnitude of vertical disparity is constant in a row. These vertical disparity patterns can be seen in Figure 11.

4.1.4 *Relative rotation about X-axis (Tilt)*

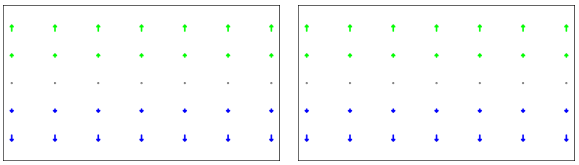
If one camera is rotated about X-axis relative to the other (as shown in Figure 12), features move up or down in the other camera's view. See Figure 13(b) and



(a) Positive zoom difference

(b) Negative zoom difference

Figure 10: Zoom error representation with left camera (thick rectangle) and right camera (dashed rectangle), and the corresponding vertical disparity pattern. The distance between dashed lines in pattern represents the magnitude of disparity.



(a) Depth=-600 (b) Depth=-400

Figure 11: Vertical disparity patterns at different depths $(u_r - u_l)$ when there is different zoom in camera pair $\alpha_f = 0.1$.

Figure 13(a) to visualise the effect. If relative tilt angle is the only error present in a stereo rig, equation 8 can be reduced to

$$v_r - v_l = -\alpha_x \hat{f}_l - \frac{\alpha_x}{\hat{f}_l} v_l v_r. \tag{12}$$

The properties of relative tilt error are: (1) the disparity map is independent of depth information, (2) the amount of vertical parallax increases or decreases from the image centre to the upper and to the lower edges and (3) if the focal length is very large, the component $-\frac{\alpha_x}{\hat{f}_l} v_l v_r$ can be ignored. In that case vertical disparity is uniform through out the image. These vertical disparity patterns can be seen in Figure 14.

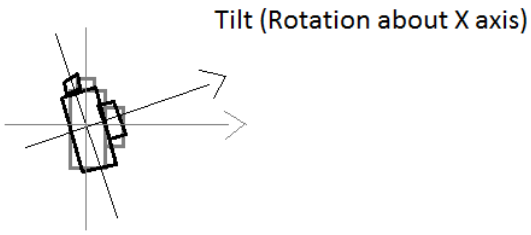
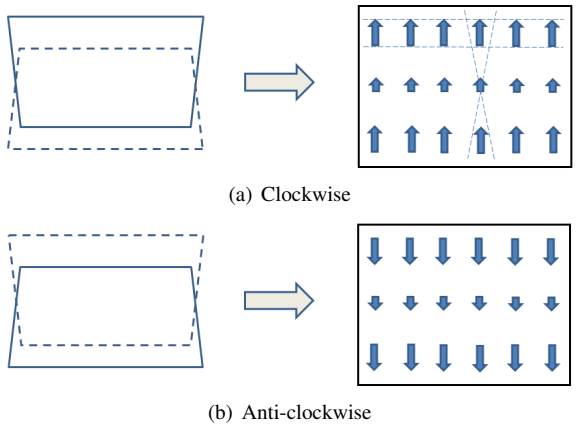


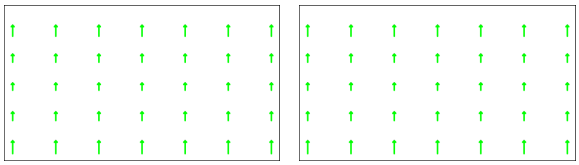
Figure 12: Cameras with relative tilt angle.



(a) Clockwise

(b) Anti-clockwise

Figure 13: Relative rotation along X-axis error of left camera (thick rectangle) and right camera (dashed rectangle) and the corresponding vertical disparity pattern. The distance between dashed lines in pattern represents the magnitude of disparity.



(a) Depth=-600 (b) Depth=-400

Figure 14: Vertical disparity patterns at different depths when the relative tilt angle between the cameras $\alpha_x = 0.1$.

4.1.5 *Relative rotation about Y-axis (Pan)*

If one camera is rotated about Y-axis relative to the other (as shown in Figure 15), it is called toed-in / toed-out setup. It results in key stone distortion. In this case, the vertical disparity can be observed clearly at the corners than at the image centre. Opposite corners exhibit similar disparity pattern while adjacent corners show opposite directions. See Figure 16(b) and Figure 16(a) to visualise the effect. If relative pan angle is the only error present in a stereo rig, equation 8 can be reduced to

$$v_r - v_l = \frac{\alpha_y}{\hat{f}_l} u_r v_l. \tag{13}$$

The properties of the relative pan angle are: (1) the disparity map is independent of depth information, (2) the direction and magnitude of the vertical parallax is the same on the opposite corners, (3) neighboring corners

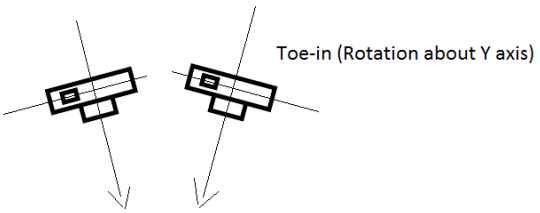
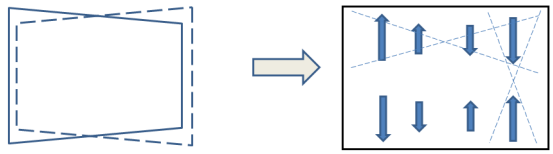
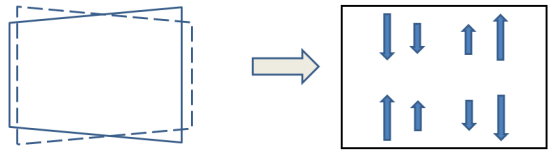


Figure 15: Cameras with relative pan angle.

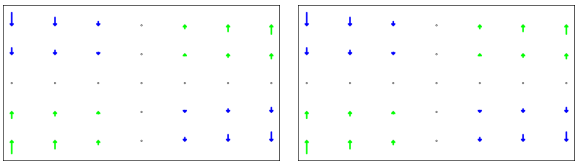


(a) Clockwise



(b) Anti-clockwise

Figure 16: Relative rotation along Y-axis error of left camera (thick rectangle) and right camera (dashed rectangle) and the corresponding vertical disparity pattern. The distance between dashed lines in pattern represents the magnitude of disparity.



(a) Depth=-600 (b) Depth=-400

Figure 17: Vertical disparity patterns at different depths when the relative pan angle between the cameras $\alpha_y = 0.1$.

differ in the direction of vertical disparity but the magnitude of the vertical disparity is the same and (4) the value of vertical disparity increases from the top edge to the bottom edge on one side and decreases on the other side of the image central column. These vertical disparity patterns can be seen in Figure 17.

4.1.6 *Relative rotation about Z-axis (Roll)*

If one camera is rotated about *Z*-axis relative to the other (as shown in Figure 18), the features on one side of the central column move up and on the other side move down. See Figure 19(a) and Figure 19(b) to visualise the effect. If relative roll angle is the only error present in a stereo rig, equation 8 can be reduced to

$$v_r - v_l = \alpha_z u_r. \tag{14}$$

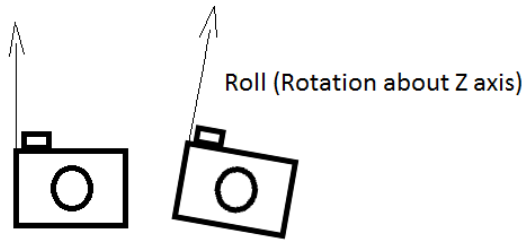
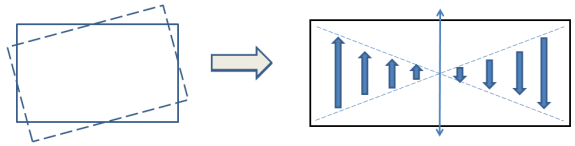
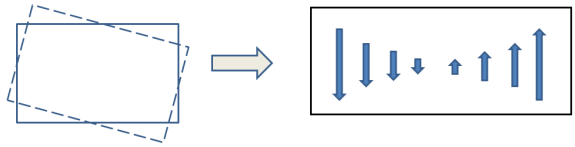


Figure 18: Cameras with relative roll angle.

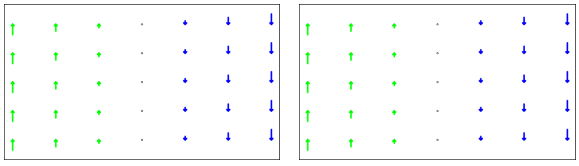


(a) Anti-clockwise



(b) Clockwise

Figure 19: Relative rotation along Z-axis error of left camera (thick rectangle) and right camera (dashed rectangle) and the corresponding vertical disparity pattern. The distance between dashed lines in pattern represents the magnitude of disparity.



(a) Depth=-600 (b) Depth=-400

Figure 20: Vertical disparity patterns at different depths when the relative roll angle between the cameras $\alpha_z = 0.1$.

The properties of the relative roll angle are: (1) the disparity map is independent of depth information, (2) the value of vertical parallax increases or decreases from the left edge to the right edge of the image (in a row) and (3) the vertical disparity pattern is symmetrical about central column of the image. These vertical disparity patterns can be seen in Figure 20.

4.1.7 *Multiple errors (Tilt and Roll)*

The patterns can be classified also for combination of the errors. Here, only the tilt and roll combination is presented. If relative tilt and relative roll are present in a stereo rig, it results in disparity patterns as shown in Figure 21. Equation 8 can be reduced to

$$v_r - v_l = \alpha_z u_r - \hat{f}_l \alpha_x - \frac{\alpha_x}{\hat{f}_l} v_l v_r. \tag{15}$$

The properties of the resulting vertical disparity map are: (1) the disparity map is independent of depth (prop-

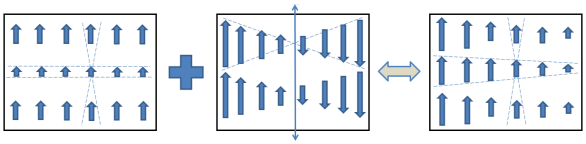


Figure 21: Resulting vertical disparity pattern when relative rotations along X-axis (tilt) and Z-axis (roll) are present simultaneously. The distance between dashed lines in pattern represents the magnitude of disparity.

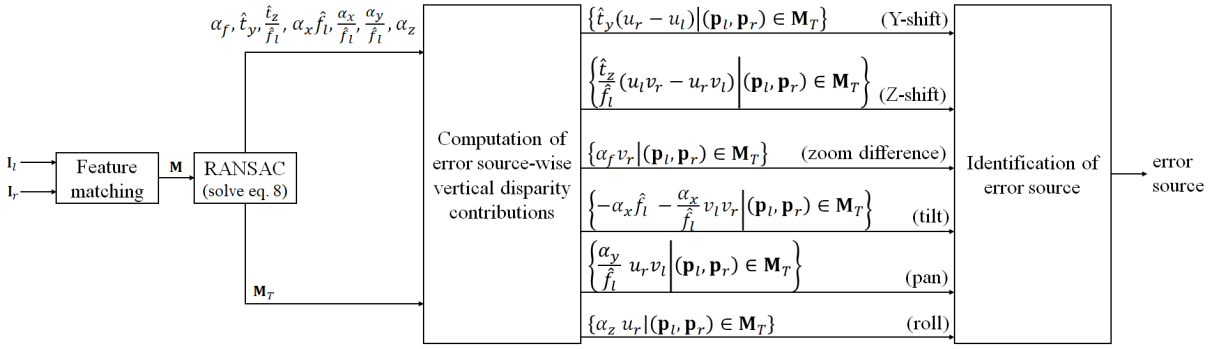
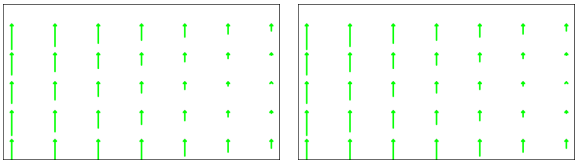


Figure 22: Block diagram showing the steps in the proposed mathematical approach. Key - $\mathbf{I}_l$: left image, $\mathbf{I}_r$: right image, $\mathbf{p}_l = (u_l, v_l)$: a point on the left image, $\mathbf{p}_r = (u_r, v_r)$: a point on the right image, $\mathbf{M}$: set of feature point matches, $\mathbf{M}_T \subseteq \mathbf{M}$: set of true feature point matches, α_f: zoom difference (focal length difference between two cameras), $\hat{t}_y$: relative shift in Y-direction, $\hat{t}_z$: relative shift in Z-direction, α_x: relative tilt, α_y: relative pan, α_z: relative roll, and $\hat{f}_l$: focal length of left camera in pixels.



(a) Depth=-600 (b) Depth=-400

Figure 23: Vertical disparity at different depths ($u_r - u_l$) when there are relative tilt and roll angles between the cameras $\alpha_x = 0.1$ and $\alpha_z = 0.1$ respectively.

erty of rotational errors), (2) the value of vertical parallax increases or decreases from the left edge to the right edge of the image (property of roll error), and (3) the value of vertical parallax increases or decreases from the image center to the upper and to the lower edges of the image (property of tilt error). These vertical disparity patterns can be seen in Figure 23.

4.2 Error identification

If there are multiple error sources present simultaneously, it is difficult to identify them from the classified patterns using the proposed classification based approach (shown in Figure 3). Moreover, there can be false matches that create false disparity at certain locations in the map. To overcome these challenges we propose a mathematical approach that solves the unknowns in equation 8. As there are 7 unknowns, namely $\alpha_f, \hat{t}_y, \hat{t}_z/\hat{f}_l, \alpha_x\hat{f}_l, \alpha_x/\hat{f}_l, \alpha_y/\hat{f}_l$ and α_z, at least 7 pairs of ($\mathbf{p}_l$, $\mathbf{p}_r$) are needed to solve for the unknowns. To increase the reliability of the solution in the presence of false feature matches, RANSAC algorithm [Fischler81] is used. In this work, RANSAC uses a threshold on Sampson distance to identify false matches. After filtering the false matches, the vertical disparity contributions of each error source are computed for each true match. The error sources that highly influence the total vertical disparity are identified and are considered as the major rig-alignment errors. Figure 22 presents the block diagram of the approach.

5 RESULTS

This section presents the experimental results of the proposed approach. According to my knowledge, this is the first work in this direction and there exists no other work to compare the performance. Consider a stereo image pair (shown in Figure 24) captured using an uncalibrated stereo rig i.e. unknown intrinsic and extrinsic parameters of the cameras. Sparse feature matches are computed using SIFT, SURF and Harris features. Figure 25 show the results of stereo matching. It can be observed that there exists false matches. After applying the RANSAC on equation 8, the false matches are filtered and the unknowns $(\alpha_f, \hat{t}_y, \hat{t}_z/\hat{f}_l, \alpha_x\hat{f}_l, \alpha_x/\hat{f}_l, \alpha_y/\hat{f}_l, \alpha_z)$ are calculated. Figure 26 shows the results of stereo matching after removing false matches. Figure 27 shows the vertical disparity at the true matched features drawn on the left image. It can be observed that the disparity increases from right to left and in the entire image the disparity is upwards. Based on the error to vertical disparity pattern classification, the pattern in the results is close to the one caused by tilt and roll (see Figure 21). To confirm this, a graph that shows the individual contributions of the possible error sources is drawn and the errors that highly influence the total vertical disparity are identified. Figure 28 shows the contributions of each possible error source. It can be observed that tilt and roll are the major alignment errors influencing the total vertical disparity.

(a) Left image (b) Right image

Figure 24: Input stereo image pair.

Figure 25: Visualisation of feature matches (using SIFT, SURF and Harris features) on the right image by drawing a line from the left image point to the right image point.

Figure 26: Feature matches after filtering false matches using RANSAC. The matched points are visualised on the right image by drawing a line from the left image point to the right image point.

6 CONCLUSION

Depending on the type of rig alignment error the vertical disparity is different in different regions of the stereo images. This paper presents a classification of possible disparity patterns to make stereographers or quality analysts identify the error source through visual inspection of the patterns in the vertical disparity map. SIFT, SURF and Harris feature detection algorithms are used to produce the map. This paper also proposed a mathematical approach that computes the contribution of each possible error to the vertical disparity. The advantage is that the approaches do not require any calibration information about the stereo rig. The possible future works include automatic adjustment of rig alignment based on the computed vertical disparity contributions and evaluation of the results using controlled experiments on a real stereo rig setup.

Figure 27: Vertical disparity map drawn on the left image. The red circles indicate the points on the left image and the yellow lines show the vertical disparity at the points.

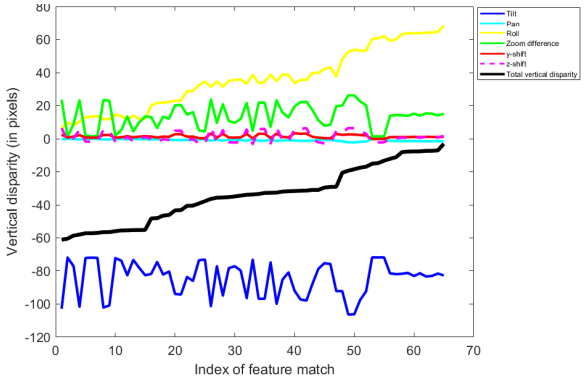


Figure 28: Contributions of possible error sources to the vertical disparity at each feature match. Total vertical disparity is also drawn. The feature matches are sorted in the ascending order of their vertical disparity. Tilt and roll contributions are highly influencing the total vertical disparity.

7 REFERENCES

[Andrea00] Andrea, F., and Emanuele, T., and Alessandro, V. A Compact Algorithm for Rectification of Stereo Pairs, ACM Machine Vision Applications, Vol.12, No.1, pp.16-22, July, 2000.

[ATSC11] Advanced Television Systems Committee (ATSC), Final Report of the ATSC Planning Team on 3D-TV, August, 2011.

[Bay08] Bay, H., and Tuytelaars, T., and Gool, L.V. SURF:Speeded-Up Robust Features, Computer Vision and Image Understanding (CVIU), Vol.110, pp.346-359, June, 2008.

[Boufama95] Boufama, B., and Mohr, R. Epipole and Fundamental Matrix Estimation using Virtual Parallax, Proceedings of the Fifth International Conference on Computer Vision (ICCV), pp.1030-1036, 1995.

[Clarke98] Clarke, T.A., and Wang, X., and Fryer, J.G. The Principal Point and CCD Cameras, The Photogrammetric Record, Vol.16, pp.293-312, Octo-

ber, 1998.

[Faugeras93] Faugeras, O. Three-Dimensional Computer Vision (Artificial Intelligence), The MIT Press, November, 1993.

[Fischler81] Fischler, Martin A., and Bolles, Robert C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography, Communications of the ACM, Vol.24, No.6, pp.381-395, June, 1981.

[Hartley04] , Hartley, R., and Andrew, Z. Multiple View Geometry in Computer Vision, Cambridge University Press, ed.2, April, 2004.

[Hartley92] Hartley, R.I. Estimation of Relative Camera Positions for Uncalibrated Cameras, Proceedings of the European Conference on Computer Vision (ECCV), pp.579-587, 1992.

[Hodges91] Hodges, L.F. Basic Principles of Stereographic Software Development, Proceedings of the SPIE (Society of Photo-Optical Instrumentation Engineers) on Stereoscopic Displays and Applications II, Vol.1457, pp.9-17, 1991.

[IJsselsteijn00] IJsselsteijn, W. A., and de Ridder, H., and Vliegen, J., Effects of stereoscopic filming parameters and display duration on the subjective assessment of eye strain, Proceedings of SPIE, Stereoscopic Displays and Virtual Reality Systems VII, vol.3957, pp.12-22, May, 2000.

[Kamgar-Parsi88] Kamgar-Parsi, B., and Eastman, R.D. Calibration of a Stereo System with Small Relative Angles, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol.51, No.1, pp.1-19, 1988.

[Knorr12] Knorr, S., and Ide, K., and Kunter, M., and Sikora, T. The Avoidance of Visual Discomfort and Basic Rules for Producing "Good 3D" Pictures, SMPTE Motion Imaging Journal, Vol.121, issue 7, pp.72-79, October, 2012.

[Lenz88] Lenz, R.K., and Tsai, R.Y. Techniques for Calibration of the Scale Factor and Image centre for High Accuracy 3-D Machine Vision Metrology, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.10, pp.713-720, 1988.

[Lowe04] Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints, International Journal on Computer Vision, Vol.60, No.2, pp.91-110, November, 2004.

[Perek16] Perek, P., and Makowski, D., and Napieralski, A. Efficient uncalibrated rectification method for stereo vision systems, MIXDES - 23rd International Conference Mixed Design of Integrated Circuits and Systems, pp. 89-92, June, 2016.

[Sammons92] Sammons, E. The World Of 3-D Movies, Delphi Publication, 1992.

[Tam11] Tam, W.J., and Speranza, F., and Yano, S., and Shimono, K., and Ono, H. Stereoscopic 3D-TV: Visual Comfort, IEEE Transactions on Broadcasting, Vol.57, pp.335-346, 2011.

[Woods93] Woods, A., and Docherty, T., and Koch, R. Image Distortions in Stereoscopic Video Systems. Proceedings of SPIE, Stereoscopic Displays and Applications IV, Vol.1915, pp.36-48, September, 1993.

[Zilly10] Zilly, F., and Muller, M., and Eisert, P., and Kauff, P. Joint Estimation of Epipolar Geometry and Rectification Parameters using Point Correspondences for Stereoscopic TV Sequences, 3D Data Processing, Visualization and Transmission (3DPVT) Conference, May, 2010.

[Zilly11] Zilly, F., and Kluger, J., and Kauff, P. Production Rules for Stereo Acquisition, Proceedings of the IEEE, Special Issue on 3D Media and Displays, Vol.99, issue 4, pp.590-606, April, 2011.

ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.27, No.2, 2019

Triplet Loss with Channel Attention for Person Re-identification

Daniel Organisciak, Chirine Riachy, Nauman Aslam, Hubert P. H. Shum†
Northumbria University
Department of Computer and Information Sciences
{daniel.organisciak, chirine.riachy, nauman.aslam, hubert.shum}@northumbria.ac.uk
† corresponding author

ABSTRACT

The triplet loss function has seen extensive use within person re-identification. Most works focus on either improving the mining algorithm or adding new terms to the loss function itself. Our work instead concentrates on two other core components of the triplet loss that have been under-researched. First, we improve the standard Euclidean distance with dynamic weights, which are selected based on the standard deviation of features across the batch. Second, we exploit channel attention via a squeeze and excitation unit in the backbone model to emphasise important features throughout all layers of the model. This ensures that the output feature vector is a better representation of the image, and is also more suitable to use within our dynamically weighted Euclidean distance function. We demonstrate that our alterations provide significant performance improvement across popular re-identification data sets, including almost 10% mAP improvement on the CUHK03 data set. The proposed model attains results competitive with many state-of-the-art person re-identification models.

Keywords

Person Re-identification, Squeeze and Excitation, Triplet Loss, Metric Learning, Siamese Network, Channel Attention, Weighted Euclidean

1 INTRODUCTION

Person re-identification (re-ID) is a core challenge within computer vision where an identity observed in one camera is required to be matched with another observation from a different viewpoint. This task has attracted a lot of interest due to the potential applications in the real-world as an increasing volume of large-scale urban surveillance data is collected.

In the past few years, convolutional neural networks (CNNs) have become ubiquitous within person re-ID due to significantly improving upon the state-of-the-art results [1, 2, 3]. Many person re-ID works make use of the standard convolutional neural networks with a classification loss [4]. More specific to re-ID, however, is the use of the triplet loss function [2, 5, 6, 7], either in place of or alongside the standard cross-entropy loss. The triplet loss, shown in Figure 1, enforces a distance margin, α, between the set of images of one person and all other images.

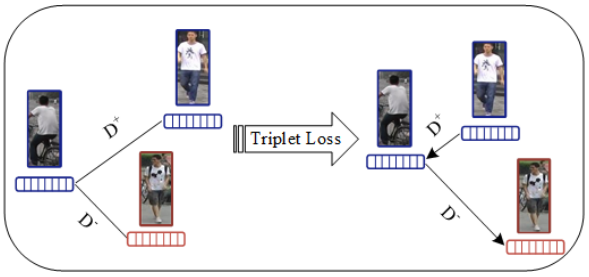


Figure 1: The triplet loss aims to reduce the distance of feature vectors from similar identities and increase the distance of feature vectors from dissimilar identities. We use channel attention in the form of squeeze and excitation units to get a better feature representation and improve the Euclidean distance by adding dynamic weights for each feature.

To date, most triplet loss works focus on mining better samples to improve the model generalisation [2, 10], or alter the loss function in order to increase the inter-class variance and decrease the intra-class variance [5, 7]. We identify two additional, under-researched lines of work to improve the triplet loss: improving the feature vectors obtained from the deep learning architecture by exploiting squeeze and excitation (SE) units, and adding dynamic weights to the distance function with which the triplet loss compares these feature vectors. We show that these alterations improve re-ID precision individually. When implemented together, these adjustments

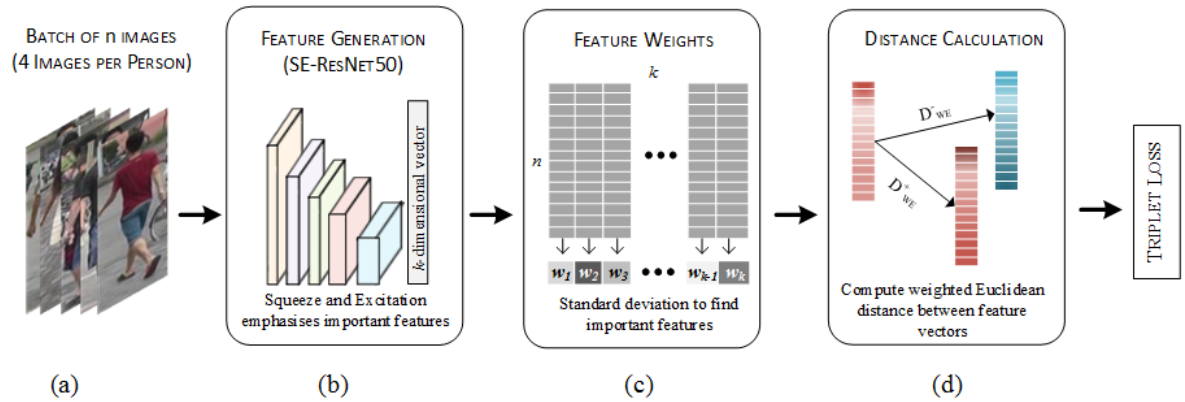


Figure 2: An overview of our architecture: (a) an input batch of n images is generated, (b) the batch is processed by SE-ResNet50 [8, 9] to generate one feature vector per image, (c) the standard deviation for each feature is computed then normalised to attain weights, (d) our improved triplet loss processes the mined triplets.

complement each other, resulting in a performance improvement of over 9% mAP on the CUHK03 data set compared to the regular triplet loss.

Distance: The triplet loss, by its nature, attempts to decrease the distance between positive pairs of images while increasing the distance of negative ones. However, to date, little research has been done to assess exactly how this distance should be formulated. The Euclidean distance has been shown to perform well within the triplet loss function, thus has not received much scrutiny. We show that adding dynamic weights to the Euclidean distance can deliver considerable benefit when applied to the task of person re-ID.

The standard Euclidean distance considers all features as equally important. As shown in Figure 2 (c), our dynamically weighted Euclidean distance assigns an importance score to each feature derived from a feature's batch-wise standard deviation. Features with higher variance are more informative, thus assist the model to distinguish between images of different identities. To conceptualise this idea, if everyone in a batch wears a plain, white t-shirt, it is impractical to consider this information for re-ID. We assess the batch-wise feature vectors for high-level features that act in this manner and diminish their importance while highlighting more useful features.

Features: We would like our backbone architecture to generate feature representations of images which can best be exploited by the dynamically weighted Euclidean distance. In order to achieve this, we use channel attention by adding SE units into our framework. These units act as weights to magnify important channels at each layer of the network while depreciating the value of less important channels. At deeper layers, these weights become more polarising to ensure salient features derived from the important channels are distinguishable from less important features.

As the less important features are mapped towards 0 by the SE units, they are more likely to have a low standard deviation and will therefore be assigned small weights by our dynamically weighted Euclidean distance.

The main contributions of this paper are as follows:

1. *Dynamically Weighted Euclidean Distance for Triplet Loss Feature Accentuation:* We introduce a weighted Euclidean distance which highlights features with high variation across the batch, in order to disregard features which are unimportant or susceptible to noise. This alone provides consistent performance improvement across all tested data sets.

2. *Feature Vector Generation with Channel Attention:* We are the first to adopt SE-ResNet 50 as the backbone architecture for the triplet loss. We demonstrate that the channel attention that SE units provide significantly boosts the performance of the triplet loss across a variety of data sets.

The rest of the paper is organised as follows: Section 2 contains an overview of work related to this paper. Section 3 details the formulation of our dynamically weighted Euclidean distance and explains how we use channel attention through SE units to boost the performance of the triplet loss. Section 4 contains our experimental results. Section 5 concludes the paper and discusses potential future directions that this work opens up.

2 RELATED WORK

Traditionally, popular methods for person re-ID comprised of two components: designing hand-crafted features [11] and learning distance metrics [12]. Hand-crafted features were required to be robust to variations in light, pose and viewpoint while using conventional distance metrics like the Mahalanobis distance [13],

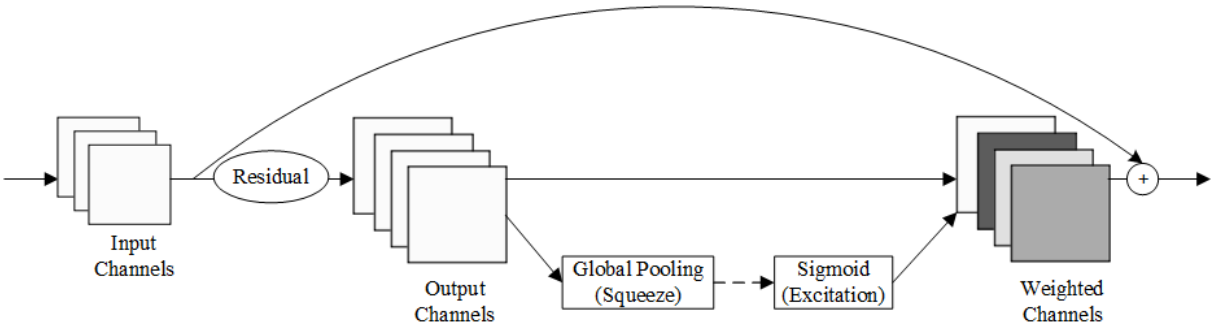


Figure 3: An overview of a ResNet block with a squeeze and excitation unit.

Bhattacharyya distance, and the l_1- and l_2-norms. In this respect, our work can be seen to be similar to this category of works, but within a deep learning context. We wish to improve the feature representations that are output by the backbone architecture, and also develop a better distance metric to compare these representations within a triplet loss setting.

2.1 Backbone Architecture

ResNet50: The majority of person re-ID works use the 50 layer variant of ResNet [9] as the backbone architecture. One possible reason for ResNet50 being ubiquitous in re-ID specifically is that mid-level features are relatively important compared with other fields. The skip connections in ResNet ensure that these important mid-level features have more presence in the features output by later levels compared with other popular backbones. Yu *et al.* [14] concatenate features from earlier ResNet layers with the final layer outputs to obtain a better representation. Zeng *et al.* [15] achieve state-of-the-art results with a hierarchical deep learning feature, which fuses features from several earlier layers, and define a new metric to best exploit this new feature.

Random Erasing [16] is a data augmentation technique that randomly removes a small area of each image in the input batch before processing them with ResNet. Sun *et al.* [17] use ResNet50 to learn discriminative features which are informed by 'parts' from the input image. Sun *et al.* [18] use Singular Vector Decomposition within ResNet to optimise the deep representation learning process. Due to its proven success, we also use ResNet50. We incorporate SE units to inform the network which channels are most important at each layer. This carries through to the output feature vector and allows us to tailor our loss function to identify the most salient features to assign more weight.

Squeeze and Excitation Networks: Squeeze and Excitation Networks [8] are frameworks that incorporate a squeeze and excitation unit at all or some layers of the architecture. Channels are obtained in a convolutional neural network for each filter learned by the CNN. Channel-wise spatial information is first 'squeezed' into per-channel descriptor to assess the relative importance of each channel. This information is then passed through a gating mechanism to 'excite' the descriptor. The original channels are then multiplied by their respective channel descriptor obtained from the squeeze and excitation process.

To date, SE units have seen limited use in person re-ID. Wang *et al.* [19] adapt SE Units as part of a fully attentional block. Li *et al.* [3] combine channel attention with spatial attention and part-wise attention to create their Harmonious Attention CNN. To the best of our knowledge, we are the first to use SE units purely as a backbone architecture to exploit important features that are generated as part of the output feature vector.

2.2 Triplet Loss

The triplet loss function has been used extensively for person re-ID due to its proven ability to attain state-of-the-art results [2, 10]. Traditional triplet models take three images as input: a query image, a positive image that has the same identity as the query, and a negative image that has a different identity to the query. A margin α is enforced to ensure a certain distance between positive and negative pairs.

Wang *et al.* [20] proposed to use the triplet loss function to learn image similarity. The triplet loss gained notoriety by significantly improving the state of the art for face verification [21, 22]. Since then, triplet loss research has typically focused on improving either the triplet mining algorithm or the loss function.

Building an effective triplet network is heavily reliant on the mining strategy. To challenge the framework to be able to handle tough cases, difficult triplets need to be mined, but choosing only the hardest triplets in the data set will result in a model that is not representative of the entire set of triplets. To strike the balance between finding difficult triplets while still generating a representative model, Hermans *et al.* [2] present *Batch Hard* mining, which selects only the hardest triplets

across each batch selected during training. In a similar manner, Almazan *et al.* [10] select triplets that start off relatively easy but get more difficult as training progresses.

Cheng *et al.* [5] introduce an improved triplet loss function that decreases the distance of images from the same identity whilst increasing the distance of images from a different identity. Chen *et al.* [7] add an additional term to the triplet loss to form a quadruplet loss. This term contains a second negative pair which helps to enlarge inter-class variations across the data set. Jiang *et al.* [23] demonstrate improved performance through adding a self-supervised attention loss to the quadruplet loss. While performance is enhanced by these works, they all focus on improving the same aspects of the triplet loss. We instead tackle the under-researched feature representation and the distance function.

3 METHODOLOGY

3.1 Triplet Loss Background

We formulate the triplet loss mathematically in order to provide motivation to investigate the distance metric and the feature representation.

We denote a triplet, $t = (x, x^+, x^-)$, where x is the query image, x^+ is a positive image, and x^- is a negative image. The triplet loss function is formulated as follows:

$$\begin{aligned}\mathscr{L}_{trp} = \sum_{t \in \mathscr{T}} \max((||f(x) - f(x^+)||_2 \\ - ||f(x) - f(x^-)||_2 + \alpha), 0),\end{aligned} \tag{1}$$

where the feature vector of an image x obtained from the convolutional neural network is denoted as $f(x)$, $\mathscr{T}$ is the set of mined triplets and $||\cdot||_2$ denotes the Euclidean distance. This loss will force negative images to be a distance of at least α away from the positive pair.

Let p be the identity of the image $x_{p,i}$ in the batch B, where $f(x_{p,i})$ is its feature vector, $p = 1, \ldots, P$ and $i = 1, \ldots, 4$. Each query image $x_{p,i}$ is paired with its hardest positive image x^+ and hardest negative image x^-, which are found via the equations:

$$x^+ = \max_{x_{q,j} \in B} (||(f(x_{p,i}) - f(x_{q,j})||_2), \text{ where } p = q, \tag{2}$$

$$x^- = \min_{x_{q,j} \in B} (||(f(x_{p,i}) - f(x_{q,j})||_2), \text{ where } p \neq q. \tag{3}$$

From equations (1) - (3), we see that obtaining the feature representation $f(x)$, and computing the distance between feature representations of any two images, $||(f(x_1) - f(x_2)||$, are essential components of the triplet loss. We focus our research on improving these two aspects.

3.2 Dynamically Weighted Euclidean Distance

Although the triplet loss has seen extensive use in person re-ID, there has been little work to deviate from the standard Euclidean distance, despite it being a crucial element of the framework. We improve it by weighting each feature based on its importance.

To calculate which features are most discriminative, we use the $n \times k$ feature matrix output from the backbone of the network to calculate the standard deviation for each feature across the batch. This is shown in Figure 2 (c). The higher the standard deviation, the more variation in that feature, and the more effective it is at helping the framework to tell people apart. These more meaningful features should thus be assigned a greater weight.

We use a softmax function regularisation on the standard deviations, then multiply by the total number of features to obtain the final weights. Overall, the weight, w_i, for the i-th feature can be calculated as

$$w_i = \text{softmax}\left(\text{s.d.}(\mathbf{F_i})\right) \times k, \tag{4}$$

where $\text{s.d.}(\cdot)$ is the standard deviation and $F \in \mathbb{R}^{n \times k}$ is the batch-wise feature matrix output by the backbone of the model with features $i = 1, \ldots, k$.

To ensure that the more important features are more prominent when calculating the distance matrix, we use the weighted Euclidean distance, D_{WE}, between two feature vectors, $\mathbf{x}$ and $\mathbf{y}$:

$$D_{WE}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{k} w_i (x_i - y_i)^2}, \tag{5}$$

where w_i are the weights and k is the length of the feature vectors.

The standard triplet loss will separate embeddings to ensure that the distance between classes is greater than the hard margin α. Because we iteratively adjust the formulation of this distance, we are able to push classes apart even further, which leads to the model better representing the data.

3.3 Channel Attention Feature Embedding

The triplet loss evaluates the distance between feature representations, thus is very dependant on the quality of the feature vectors that are generated by the network. Furthermore, we would like these feature vectors to possess information which can be exploited by our dynamically weighted Euclidean distance.

We concentrate on improving the feature representations themselves by utilising channel attention via SE blocks [8] within ResNet50.

An SE unit is a mechanism that performs feature recalibration within the framework utilising it. By doing so, it selects features that are the most informative to the framework and accentuates them, while diminishing the importance of less useful features. These informative features then allow the re-id framework to create a better embedding which is more effective at separating classes.

In this regard, the SE unit acts as a process to determine the weight of each channel at each layer of the model, similarly to how our dynamically weighted Euclidean distance performs. The SE units perform different roles throughout the network, getting more polarising at deeper layers. As a consequence, unimportant channels are mapped near to 0 in the final block of ResNet, which has a large effect on the output feature representation of each image.

Note that as unimportant features are mapped towards 0 throughout the network, they will typically have a low standard deviation. On the contrary, important features will be less impacted by the SE units and are therefore more likely to have a higher standard deviation. This means that our dynamic weights will be much more likely give a large weight to features that are computed to be important by SE units, while still being able to identify features with high variance even though they are not determined to be salient by the network.

As we show in Figure 3, the unit first squeezes the channel-wise spatial information into a channel descriptor via Global Average Pooling. Formally, given a channel $u \in \mathbb{R}^{H\times W}$, we squeeze it to obtain its channel descriptor, c, as follows:

$$c = \text{squeeze}(u) := \frac{1}{H\times W}\sum_{i=1}^{H}\sum_{j=1}^{W} u_{ij}, \tag{6}$$

where H and W are the height and width of the channel respectively. These channel descriptors form a vector $\mathbf{z} = [c_1, \ldots, c_C]$ where C is the total number of channels.

Next, in order to calculate the channel-wise dependencies, this statistic needs to be excited. To achieve this, a simple gating mechanism with a sigmoid activation function is employed similarly to what is used within many spatial attention methods. The vector of squeezed channel descriptors $\mathbf{z}$ is passed through a dimensionality-reduction fully connected layer, a ReLU and then a dimensionality-increasing fully connected layer. This is then processed by a sigmoid activation to obtain the excited channel descriptors.

Formally, this excitation is written as:

$$\mathbf{s} = \text{excite}(\mathbf{z}) := \sigma\left(\mathbf{W_2}\delta(\mathbf{W_1}\mathbf{z})\right), \tag{7}$$

where $\mathbf{W_1} \in \mathbb{R}^{\frac{C}{r}\times C}, \mathbf{W_2} \in \mathbb{R}^{C\times\frac{C}{r}}$ are the parameters of the dimensionality-reduction and dimensionality-increasing layers respectively, δ is the ReLU function and σ is the sigmoid activation function.

We show in Section 4 that this adaptation alone vastly improves the performance of the triplet loss on multiple data sets.

4 EXPERIMENTS

4.1 Evaluation Protocol

We perform experiments on the two most commonly used data sets to evaluate deep learning methods for person re-ID, CUHK03 [31] and Market-1501 [32]. In addition, we also provide results on VIPeR [33] to demonstrate that our method can considerably improve performance even on very small data sets, which many deep learning frameworks struggle on. We report the mean average precision (mAP) and top-1 matching rate (rank-1) scores for CUHK03 and Market-1501, and rank-1, rank-5 and rank-10 scores for VIPeR.

The rank-x matching rate is defined as the percentage of query images with a correct match within the highest x ranks. The precision, P_x, of a framework at rank x is written as

$$P_x = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}. \tag{8}$$

To obtain the average precision for a given query, the average of the precision scores at each *true* positive in the ranking list is calculated. That is,

$$AP = \frac{1}{N}\sum_{i=1}^{N} P_i^+, \tag{9}$$

where N is the number of true positives in the gallery and P_i^+ denotes the precision at the i-th true positive in the ranking list. The mAP is then calculated by taking the mean of the average precision of all images in the query set.

Many works boost the performance of their framework with a post-processing technique such as re-ranking [34] or a data augmentation procedure like random erasing [16]. Although our results would improve, we do not use re-ranking or random erasing in any of our experiments as it does not help to evaluate the core performance of the network.

Throughout our experiments on CUHK03 and Market-1501, we use a batch size of $n = 96$ with four images per person, while on VIPeR we use a batch size of $n = 32$ with two images per person. The feature representations, $f(x)$, contain $k = 2048$ features in all of our experiments. We fix the margin $\alpha = 0.3$.

Note that, to fairly compare the effect of the dynamically weighted Euclidean distance, we use it within the triplet loss function but don't apply it during the mining phase.

All experiments are performed on a single NVIDIA GeForce GTX 1070 Ti GPU. Our model takes around 1

Comparison with baseline methods							
Data Set	CUHK03		Market-1501		VIPeR		
Method	mAP	rank-1	mAP	rank-1	rank-1	rank-5	rank-10
ResNet50	26.3	26.6	68.3	85.8	11.1	32.6	44.0
SE-ResNet50	37.8	38.6	72.4	87.9	17.4	40.8	51.3
TriNet*	48.8	51.4	67.9	83.4	38.3	67.7	80.4
Ours: DWE TriNet	54.8	56.1	69.7	84.2	39.2	73.7	83.2
Ours: SE TriNet	52.9	54.7	73.1	**88.1**	40.2	69.6	80.4
Ours: SE+DWE TriNet	**58.2**	**60.7**	**74.2**	88.0	**44.9**	**75.6**	**86.1**

Table 1: Comparison with baseline methods. *Trained with a hard margin $\alpha = 0.3$.

hour, 30 minutes to train on Market-1501 and around 35 minutes to train on CUHK03. We note that the system can further be optimised by tuning hyperparameters.

CUHK03: The CUHK03 data set contains 14297 bounding boxes of 1467 persons, with 767 identities used for training and 700 identities used for testing.

CUHK03 has two evaluation settings: the *labelled* setting contains bounding boxes that are manually annotated, and the *detected* setting contains bounding boxes that are automatically detected. We perform all of our experiments on the detected setting as it is a more realistic setup, which contains misplaced bounding boxes making the problem more challenging. It is more similar to what we would expect when applying re-ID to real-world tasks.

Market-1501: The Market-1501 data set contains images of 1501 people from six different cameras. The data set is split into 12936 images of 751 identities for training and 19732 of 750 identities for testing. We use the single query setting throughout all of our experiments.

VIPeR: The VIPeR data set consists of 632 pedestrians captured by two cameras. Deep learning methods typically do not report performance for this data set so we train all models ourselves with a batch size of 32. As VIPeR only contains one image per person in each camera, we replace the mAP metric with rank-5 and rank-10 precision scores.

CUHK03 (767/700) split		
Method	mAP	rank-1
DPFL [4]	37.0	40.7
SVDNet[18]	37.2	41.5
HACNN [3]	38.6	41.7
MLFN [24]	47.8	52.8
TriNet [2]	48.8	51.4
TriNet + RE [16]	50.7	55.5
DaRe [25]	51.3	55.1
PCB* [17]	57.5	63.7
HPM* [26]	57.5	63.9
MGN* [27]	**66.8**	**66.0**
Ours: DWE TriNet	54.8	56.1
Ours: SE TriNet	52.9	54.7
Ours: SE+DWE TriNet	58.2	60.7

Table 2: Comparison with baseline methods on the CUHK03 data set with the new split. *Use part-based information

Market-1501 (Single Query)		
Method	mAP	rank-1
DeepTransfer [28]	65.5	83.7
JLML [29]	65.5	85.1
TriNet [2]	67.9	83.4
TriNet + RE [16]	71.3	87.1
DaF [30]	72.4	82.3
DPFL [4]	73.1	88.9
HACNN [3]*	75.7	91.2
PCB* [17]*	**81.6**	**93.8**
Ours: DWE TriNet	69.7	84.2
Ours: SE TriNet	73.1	88.1
Ours: SE+DWE TriNet	74.2	88.0

Table 3: Comparison with baseline methods on the Market-1501 data set with the single query setting. For fair comparison, we don't include results which use re-ranking. *Use part-based information

4.2 Comparison with Baseline Methods

We present our results with the baseline methods in Table 1. We select the baselines as ResNet50 [9] and SE-ResNet50 [8] with a cross-entropy loss, and TriNet [2] as our model is comprised of these elements. All triplet loss models in Table 1 are trained with the hard margin $\alpha = 0.3$ for direct comparison.

We comprehensively outperform baseline methods across all data sets. In particular, on CUHK03, we enhance the mAP of the triplet loss by 9.4% and the rank 1 performance by 9.3%. We also demonstrate considerable performance improvement on Market-1501 and VIPeR. The results show that both elements of our framework provide a significant contribution to enhance the re-ID precision.

4.3 Comparison with State of the Arts

We further compare with state-of-the-art models (without re-ranking or random erasing) on the selected three

data sets. In particular we note that our simple alterations are enough to give us the second highest mAP score of any core framework on the CUHK03 data set. We also notice that the state-of-the-art deep learning methods struggle to compete with ours on a small data set such as VIPeR, which demonstrates the robustness of our model.

CUHK03: Our results on the CUHK03 data set can be found in Table 2. It can be observed that the weighted Euclidean significantly boosts the performance on CUHK03.

We attain the second highest performance across all models on mAP. Our simple alterations are shown to outperform very sophisticated, state-of-the-art models that exploit spatial attention. In particular, we outperform the state-of-the-art methods PCB [17] and HPM [26]. The only method that exceeds ours, MGN, is heavily engineered. It takes different sized portions of the original image as input, which has been shown by multiple works to substantially improve performance. We note that (i) we can add this technique to our framework, (ii) they use a triplet loss in their model, which could be improved by adopting our formulation.

The most appropriate state-of-the-art method from Table 2 for comparison is Random Erasing [16], as it has become one of the most popular techniques within re-ID and also uses a triplet loss. Our method comprehensively outperforms it, improving on its rank-1 accuracy by 10%. We further note that even if we keep the backbone architecture as ResNet50, simply changing the Euclidean distance function to our dynamically weighted Euclidean distance boosts performance more than Random Erasing. This further demonstrates the significance of the enhancements we have implemented and that the distance formulation is a crucial component which should not be overlooked when developing a triplet loss framework.

Market-1501: The results on the Market-1501 data set are presented in Table 3. We see that including squeeze and excitation blocks within the backbone architecture and adding dynamic weights into the Euclidean distance both enhance the framework. Our modified framework exceeds many state-of-the-art methods.

We note that although DPFL [4] and HACNN [3] beat us on Market-1501, their results on CUHK03 are much weaker, which indicates their models are heavily optimised towards the Market-1501 data set and not capable of generalising well. PCB [17] uses a part-based method which, as previously discussed, substantially improves performance and is compatible with our framework.

VIPeR: We outperform the state-of-the-art deep learning methods by 3.1% on the rank-1 matching rate. This demonstrates that our enhancements are very robust, even on data sets that do not have enough data for deep learning. In particular, we see that methods such as MLFN [24], despite performing well on popular deep learning data sets, do not have the ability to generalise as well as ours.

VIPeR			
Method	R1	R5	R10
MLFN [24]	28.2	50.9	62.3
TriNet [2]	38.3	67.7	80.4
PCB* [17]	41.1	70.3	84.5
TriNet + RE [16]	41.8	71.2	83.5
Ours: DWE TriNet	39.2	73.7	83.2
Ours: SE TriNet	40.2	69.6	80.4
Ours: SE+DWE TriNet	**44.9**	**75.6**	**86.1**

Table 4: Comparison with popular deep learning methods on the VIPeR data set. *Use part-based information

5 CONCLUSION

In this paper, we have evaluated the effects of feature saliency on the triplet loss function. We achieved this in two different ways: via assigning dynamic weights into the distance function used by the triplet loss, and by incorporating a backbone architecture with channel attention to emphasise important features throughout training. We demonstrate that both alterations alone boost performance of the triplet loss and complement each other for a significant improvement in precision when used together.

It has been shown recently that spatial attention or part-based understanding can dramatically improve the performance of re-ID frameworks. Our method is complementary to these part-based approaches, in the sense that we can apply our weighted Euclidean distance to the part-based feature vector obtained from their framework. One of our future directions is to use spatial attention to improve the selected parts that are fed into these systems before processing them with the improvements that we have described in this paper.

This paper demonstrates that using a simple mechanism to determine distance function weights works very well. More sophisticated strategies such as learning the weights concurrently with the feature representations could be adopted and are planned as future work.

ACKNOWLEDGEMENT

This project was supported in part by the Royal Society (Ref: IES\R2\181024).

6 REFERENCES

[1] L. Zheng, Y. Yang, A. G. Hauptmann, Person Re-identification: Past, Present and Future (8) 1–20. arXiv:1610.02984.

[2] A. Hermans*, L. Beyer*, B. Leibe, In Defense of the Triplet Loss for Person Re-Identification, arXiv preprint arXiv:1703.07737.

[3] W. Li, X. Zhu, S. Gong, Harmonious attention network for person re-identification, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[4] Y. Chen, X. Zhu, S. Gong, Person re-identification by deep learning multi-scale representations, in: The IEEE International Conference on Computer Vision (ICCV) Workshops, 2017.

[5] D. Cheng, Y. Gong, S. Zhou, J. Wang, N. Zheng, Person re-identification by multi-channel parts-based cnn with improved triplet loss function, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[6] W. Chen, X. Chen, J. Zhang, K. Huang, A multi-task deep network for person re-identification, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI'17, 2017, pp. 3988–3994.

[7] W. Chen, X. Chen, J. Zhang, K. Huang, Beyond triplet loss: A deep quadruplet network for person re-identification, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[8] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[9] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, arXiv preprint arXiv:1512.03385.

[10] J. Almazan, B. Gajic, N. Murray, D. Larlus, Re-ID done right: towards good practices for person re-identificationarXiv:1801.05339, doi:10.1109/ICCV.2013.228.

[11] C. Riachy, A. Bouridane, Person re-identification: Attribute-based feature evaluation, SAMI 2018 - IEEE 16th World Symposium on Applied Machine Intelligence and Informatics Dedicated to the Memory of Pioneer of Robotics Antal (Tony) K. Bejczy, Proceedings 2018-February (2018) 85–90. doi:10.1109/SAMI.2018.8323991.

[12] W.-S. Zheng, G. Shaogang, X. Tao, Person re-identification by probabilistic relative distance comparison, Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on (2011) 649–656doi:10.1109/cvpr.2011.5995598.

[13] P. M. Roth, M. Hirzer, M. Köstinger, C. Beleznai, H. Bischof, Mahalanobis Distance Learning for Person Re-identification, Springer London, London, 2014, pp. 247–267. URL https://doi.org/10.1007/978-1-4471-6296-4_12

[14] Q. Yu, X. Ching, Y.-Z. Song, T. Xiang, T. M. Hospedales, The Devil is in the Middle: Exploiting Mid-level Representations for Cross-Domain Instance Matching (3). arXiv:arXiv:1711.08106v2.

[15] M. Zeng, C. Tian, Z. Wu, Person Re-identification with Hierarchical Deep Learning Feature and efficient XQDA Metric (2018) 1838–1846doi:10.1145/3240508.3240717.

[16] Z. Zhong, L. Zheng, G. Kang, S. Li, Y. Yang, Random Erasing Data AugmentationarXiv:1708.04896.

[17] Y. Sun, L. Zheng, Y. Yang, Q. Tian, S. Wang, Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline), in: The European Conference on Computer Vision (ECCV), 2018.

[18] Y. Sun, L. Zheng, W. Deng, S. Wang, Svdnet for pedestrian retrieval, in: The IEEE International Conference on Computer Vision (ICCV), 2017.

[19] C. Wang, Q. Zhang, C. Huang, W. Liu, X. Wang, Mancs: A multi-task attentional network with curriculum sampling for person re-identification, in: The European Conference on Computer Vision (ECCV), 2018.

[20] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, Y. Wu, Learning fine-grained image similarity with deep ranking, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[21] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[22] O. M. Parkhi, A. Vedaldi, A. Zisserman, Deep face recognition, in: Proceedings of the British Machine Vision Conference (BMVC), 2015.

[23] M. Jiang, Y. Yuan, Q. Wang, Self-attention Learning for Person Re-identification, Bmvc.

[24] X. Chang, T. M. Hospedales, T. Xiang, Multi-level factorisation net for person re-identification, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[25] Y. Wang, L. Wang, Y. You, X. Zou, V. Chen, S. Li, G. Huang, B. Hariharan, K. Q. Weinberger, Resource aware person re-identification across mutliple grane resolutions, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[26] Y. Fu, Y. Wei, Y. Zhou, H. Shi, G. Huang,

X. Wang, Z. Yao, T. Huang, Horizontal pyramid matching for person re-identification, AAAI.

[27] G. Wang, Y. Yuan, X. Chen, J. Li, X. Zhou, Learning discriminative features with multiple granularities for person re-identification, in: Proceedings of the 26th ACM International Conference on Multimedia, MM '18, 2018, pp. 274–282.

[28] M. Geng, Y. Wang, T. Xiang, Y. Tian, Deep Transfer Learning for Person Re-identificationarXiv:1611.05244.

[29] W. Li, X. Zhu, S. Gong, Person re-identification by deep joint learning of multi-loss classification, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI'17, 2017, pp. 2194–2200.

[30] R. Yu, Z. Zhou, S. Bai, X. Bai, Divide and Fuse: A Re-ranking Approach for Person Re-identification 1–13arXiv:1708.04169.

[31] W. Li, R. Zhao, T. Xiao, X. Wang, Deepreid: Deep filter pairing neural network for person re-identification, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[32] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, Q. Tian, Scalable person re-identification: A benchmark, in: Computer Vision, IEEE International Conference on, 2015.

[33] D. Gray, H. Tao, Viewpoint invariant pedestrian recognition with an ensemble of localized features, in: D. Forsyth, P. Torr, A. Zisserman (Eds.), Computer Vision – ECCV 2008, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 262–275.

[34] Z. Zhong, L. Zheng, D. Cao, S. Li, Re-ranking person re-identification with k-reciprocal encoding, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.