Caarvida: Visual Analytics for Test Drive Videos

Alexander Achberger
alexander.achberger@daimler.com
Mercedes-Benz AG
Stuttgart, Germany

René Cutura
rene.cutura@tuwien.ac.at
University of Vienna
Vienna, Austria

Oguzhan Türksoy
oguzhan.tuerksoy@daimler.com
Mercedes-Benz AG
Stuttgart, Germany

Michael Sedlmair
michael.sedlmair@visus.uni-stuttgart.de
University of Stuttgart
Stuttgart, Germany

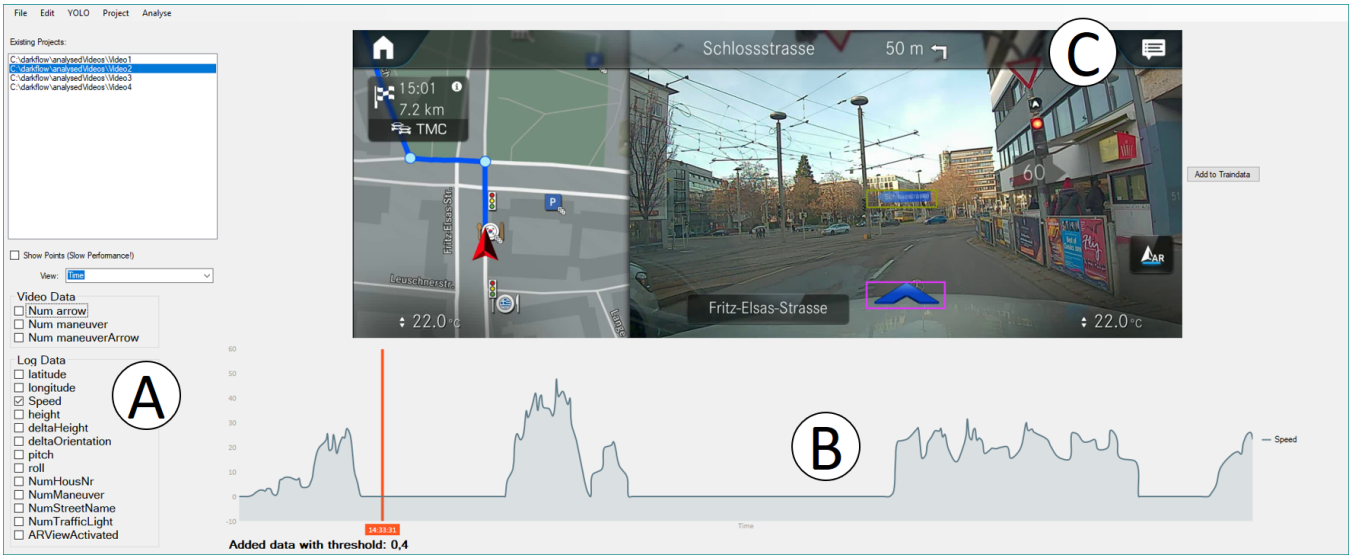

Figure 1: The *Training Data Acquisition Tool* with a combobox area (A) to select the data that is plotted in the line chart (B). A video image (C) that is selected from the line chart and bounding boxes of the detected objects from the object detection.

ABSTRACT

We report on an interdisciplinary visual analytics project wherein automotive engineers analyze test drive videos. These videos are annotated with navigation-specific augmented reality (AR) content, and the engineers need to identify issues and evaluate the behavior of the underlying AR navigation system. With the increasing amount of video data, traditional analysis approaches can no longer be conducted in an acceptable timeframe. To address this issue, we collaboratively developed Caarvida, a visual analytics tool that helps engineers to accomplish their tasks faster and handle an increased number of videos. Caarvida combines automatic video analysis with interactive and visual user interfaces. We conducted two case studies which show that Caarvida successfully supports domain experts and speeds up their task completion time.

AVI '20, September 28-October 2, 2020, Salerno, Italy

ACM ISBN 978-1-4503-7535-1/20/09…$15.00
https://doi.org/10.1145/3399715.3399862

CCS CONCEPTS

• **Human-centered computing** → **Visualization systems and tools**; • **Computing methodologies** → *Graphics systems and interfaces.*

KEYWORDS

visual analytics, human computer interaction, information visualization, object detection, automotive

ACM Reference Format:
Alexander Achberger, René Cutura, Oguzhan Türksoy, and Michael Sedlmair. 2020. Caarvida: Visual Analytics for Test Drive Videos. In *International Conference on Advanced Visual Interfaces (AVI '20), September 28-October 2, 2020, Salerno, Italy.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3399715.3399862

1 INTRODUCTION

Visualization design studies are an essential part of visualization research. Design studies are defined as projects in "which visualization researchers analyze a specific real-world problem faced by domain experts, design a visualization system that supports solving this problem, validate the design, and reflect about lessons learned in order to refine visualization design guidelines" [12]. In this work, we contribute a design study in cooperation with automotive engineers. The problem we want to solve is related to automotive video data analysis. These videos are recorded by the front cameras of cars and annotated with driving specific augmented reality content through an AR navigation system. One example is an arrow that points into the street to which the driver has to turn into (Fig. 1). The task is identifying issues and analyzing behavior in the augmented reality navigation system by analyzing the videos.

Video analysis is a very challenging task due to the high amount of information to memorize. In our case, the domain experts struggle to analyze the amount of videos in an acceptable time. To date, they have used ordinary video players to analyze their videos, meaning they cannot concentrate on multiple videos at once and must repeat tasks for multiple videos. Experts also spend too much time determining if a video is relevant for their current analysis and the time spent on analysis is proportional to the video duration. An automatic solution is not possible, because the errors and the behavior they analyze are ill-defined and subjective. Occasionally, new types of errors occur that the system must know. It is difficult to define errors, as they depend on the perception of the observer. Additionally, it is possible that the graphics, textures, and behavior of the AR navigation system will change during development. One example of an actual error is that in some cases the maneuver arrow object is too high above the street after driving uphill for a while.

We want to reduce the time experts have to spend on the aforementioned matter, so they can handle the amount of video data. After completing a task once, they should be able to quickly repeat it in other videos.

To address these issues, we engaged in a six-month iterative design process with three automotive engineers. The result is a visual analytics tool named Caarvida. Its core idea is to accomplish a given task for the first video manually with the support of interactive visual user interfaces, then complete it semi-automatically for subsequent videos by defining the task relevant scene with the used data.

We show how Caarvida speeds up the analysis for multiple videos through a quantitative analysis. Afterwards, we present two usage scenarios to demonstrate how Caarvida works. Through a problem abstraction, we discuss how Caarvida can potentially be transferred to other application areas such as sports analytics. Our main contribution is a visualization design study on augmented reality video analysis in the domain of automotive engineering, including requirement analysis, design of Caarvida, its evaluation in a quantitative analysis, and two usage scenarios.

2 RELATED WORK

We review related work in visual analysis of video data and automotive related data.

2.1 Visual Analysis of Video Data

As the volume of available video data has increased in recent decades, so too has the interest in visual analysis. One important area is video visualization aiming to summarize videos through visualization. Borgo et al. [1] provide a good overview of review methods for abstract visual representations and summaries of videos, in order to find important events and features in videos. Parry et al. [8] use storyboards to summarize Snooker matches. They focus on the challenges of event selection and event illustration. Event visualization is also often applied to for traffic monitoring. Video storyboards require a highly application-dependent event classification and many application specific semantics to be encoded in a system. Such an event classification would not work in the application specific semantics of the subject work because they are too complex. Medioni et al.[6] present a system that analyzes the behavior of moving objects from a video stream of an airborne moving platform.

Another domain where visual analysis of video data is used is in sports. Takahashi et al. [16] use a 'video poster' created from the meta-data and keyframes of large sports videos in order to summarize them. Stein et al. [4] present a visual analytics system for analyzing rugby games. They extract the player's trajectories from the video and allow sketch-based queries of their trajectories. The results of the queries are extracted candidate fragments based on their trajectories with multiple attributes, for example curvature or tortuosity. These attributes are visualized in a parallel coordinates plot. Stein et al. [14] present a visual analytics tool that combines video with movement data of soccer games. The tool visually highlights extracted events of the game to support the analyst explaining the player's behavior, for example highlight players who show a reaction to a pass. Vu et al. [20] introduce another visual analytics tool for soccer videos that helps analysts to understand formation changes.

These tools analyze videos recorded by stationary cameras to make it easier to extract information from the videos and compare different scenes within them. The visual interactive interfaces used in these approaches are highly application-dependent and would not work here.

2.2 Visual Analysis of Car Related Data

Throughout the recent decades, the number of sensors and amount of software in cars has increased. Consequently, the data collected from cars has also grown rapidly. These automotive data are very different and extensive. For example, there are sensor data, CAD data, navigation data, flow data and communication data of running systems. To visualize and gather insights into this amount of data is challenging. Sedlmair et al. [11] introduce Cardiogram, a visual analytics system that helps automotive engineers debug recorded messages from in-car communication networks. These data often consist of text files with approximately 50 million messages for an one-hour test drive. Cardiogram uses a data pre-processing approach to automatically reduce complexity based on the domain knowledge of the engineers. Tonnis et al. [17] present a system that can visualize spatial sensor data and geometric models that highlight recognized objects. Spretke et al. [15] focus on the analysis of vehicles' locations and road elevations data. They developed an

approach that uses an interval-based Parallel Coordinates visualization in order to find representative driving profiles. All these works analyze different data with different tasks. Their analysis is highly data and task dependent and would not work in our use case.

3 PROBLEM CHARACTERIZATION

In the following, we explain the knowledge we gathered from the domain experts via interviews and observations. First, we present experts' tasks and how they are currently accomplished. We then explain the data and how they are prepared. From our interviews, we derive the design requirements that are necessary to achieve the goals of the experts.

3.1 Understanding Task

The high-level task of the domain experts is enhancing the AR navigation system and to ensure its quality. Therefore, they must test the system in realistic environments. They obtain the video from test drives by recording the display video that the driver sees, as well as corresponding log data.

After receiving the data, the domain experts have two types of tasks, (1) *formative analysis* tasks and (2) *change analysis* tasks. In the following, we describe the tasks of these two types:

Formative Analysis:

- **T-error Find Errors:** The most important task is to find errors in the AR navigation system that could confuse the user or induce an incorrect navigation maneuver. An example is an arrow for turning right being rendered too high above the road.
- **T-improv Identify Room for Improvement:** This task is defined by reviewing specific situations and deciding wether they can improve the behavior of the AR navigation system at that moment, for instance if the animation speed of the arrow still appropriate when turning with a high speed.

Change Analysis:

- **T-data New Data:** It is important to check the impact of new or changed data, such as the use of new sensor data, to ensure the system uses this data correctly.
- **T-softw New Software:** Due to the fact that the AR navigation system is still in development, there are many changes in the software that have to be inspected, such as a new feature showing additional information.

3.2 Understanding Data

As described in the previous section, all data are collected while driving. One part of the data is the video that shows the result of the AR navigation system. These videos have a duration of 20 to 30 minutes in average with a maximum duration of 1 hour and a minimum duration of 2 minutes. Currently, automotive engineers receive approximately 10 videos per week, but they will receive more videos from an automatic video recording system in the future. The domain experts do not have access to the rendering process while driving, so they only record the already rendered video that is shown to the driver. They do not have the information regarding where the renderer placed the objects of the AR navigation system. Logging this information would cost too much performance and influence the system's output.

The other part is the log data. Those are a temporally ordered list of all messages from all running systems, including the AR navigation system. Log data can become extremely large due to the duration of the test drive. For example, resulting data consists of the position and orientation of the car, the position of navigation objects from the AR navigation system, and text. The logged positions of the navigation objects are latitude and longitude value where the AR navigation system positions objects on the map. As mentioned before, no render information of these objects is logged.

3.3 Design Requirements

We analyzed the current workflow and the domain experts' problems via interviews and observations of their daily work over several weeks. In their current workflow, domain experts analyze videos successively to accomplish their tasks, and analyzing videos simultaneously is impossible for them. They use ordinary media players to watch the videos and jump via a video's timeline in an arbitrary manner to find the relevant scene. To analyze the videos with log data information, they must switch between tools because they do not have a tool that combines both data. We observed multiple problems with their current workflow. The domain experts take too long to analyze videos because they analyze videos in succession. Also, the longer the video, the more time they need. After completing a task for one video, they repeat the same task for the next videos. In most cases, they watch many irrelevant scenes because they do not know where to find the relevant scenes. They do not know which data from the log file belongs to which video frame, so they analyze both data sources separately. Based on these problems and the described tasks, we define the following design requirements, which we split in two types: analysis requirements and data preparation requirements:

Data Preparation Requirements:

- **R-acqui Data Acquisition:** It should be possible to find objects and its image positions that are relevant for the tasks in the videos. In most of the tasks, the experts need to observe the behavior of an object, so it should be possible to find its occurrence and position in the data.
- **R-combi Combine Data:** Caarvida should be able to combine video data with the corresponding log data automatically. So the domain experts know what data are needed to accomplish their tasks.
- **R-adapt Adaptivity:** Our tool should still be applicable when the data changs or there are new objects added to the AR navigation system. This is essential, because the AR navigation system is still in development. Therefore, it is feasible that there will be new objects or that the data or behavior may change.

Analysis Requirements:

- **R-time Time Reduction:** The amount of time the domain experts spend analyzing multiple videos for one task should be reduced. This is the most important requirement, because the biggest problem is, repeating a task for multiple videos.

- **R-overv Overview:** Domain experts should be able to quickly gather an overview of the whole data. In many cases, domain experts must determine if the current video has scenes relevant to the task or if it can be skipped. For example, if they want to identify the maneuver object on a highway, they can skip videos that do not contain highway scenes.
- **R-repet Repetitive:** Caarvida should allow the domain experts to quickly repeat tasks for subsequent videos. They must perform one task for many videos, so it should be possible to quickly repeat the task after accomplishing it the first time.
- **R-task Task Depending Data:** Domain experts should be able to display only the data that are necessary for their current task. In most of the described tasks, irrelevant parts of the data increase the complexity.

4 CAARVIDA

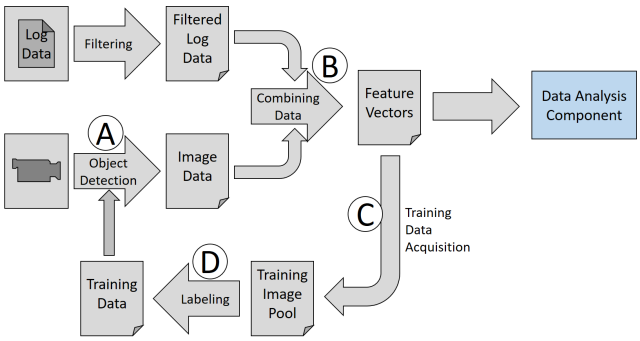

Figure 2: The data preparation process that acquires additional information from the videos via object detection (A), combines these data with the filtered log data and stores them as high dimensional data (B), acquires new training images for object detection improvement or new classes (C) and labels the training images (D).

In the following, we introduce the results of our conducted design study, a visual analytics tool called Caarvida. Its core idea is split in two steps: (1) gather and combine all data needed to find task-relevant images, find these images, and accomplish the task with them, and (2) create a query with the task relevant data in order to find the images automatically next time. For example, if the task is to check the position of maneuver objects while driving a specific slope, first find the correct slope and number of maneuver objects that define the scene which contains the task relevant images. Second, save them in a query to automatically find the images in the next videos. Based on these two steps, we designed Caarvida as two major software components: a *Data Preparation Component* and a *Data Analysis Component*.

4.1 Data Preparation Component

In the *Data Preparation Component*, we developed multiple tools and methods to efficiently generate the necessary data which is able to define task depending scenes in the *Data Analysis Component*. Its input are the log data and the videos, where additional data is acquired. The entire process is illustrated in Fig. 2.

(A) Object Detection. To be able to define scenes that contain the task dependent images, it is essential to find the important objects in the videos (R-acqui). There are many approaches for this, such as semantic segmentation, instance segmentation, or object detection, Parekh et al. [7] give a overview. The first two annotate each pixel to an object in an image, which is useful but also requires training data annotated at pixel level. Therefore, we choose object detection. We use Yolo [9] because of its performance (R-time) and its high precision. We took the implementation from Trieu [18]. The training data are manually generated from the test drive videos with the support of the *Training Data Acquisition Tool* and the *Label Tool*. Fig. 1C shows an example of detected objects.

(B) Combining Data Method. To define a task dependent scene, it is necessary to connect the images to their corresponding log data (R-combi). Combining data is highly application dependent, so we developed our own *Data Combining Method* that automatically extracts the time of each image, matches the time stamps between the images and the log data, and connects them accordingly. The combined result are high dimensional feature vectors F_j where $j \in [1, ..., N]$ and N is the amount of data and every feature of F_j is $f_i \in [0, 1]$ and $i \in [1, ..., M]$ where M is the number of dimensions/attributes of the data. Fig. 1A shows the list of all extracted features.

(C) Training Data Acquisition Tool. It is a challenge to develop a general analysis tool for accomplishing the domain experts' tasks because the AR navigation system is still under development. Therefore, Caarvida has to be able to handle system changes and new data to ensure it is still usable in the future (R-adapt). To solve this issue, we developed a *Training Data Acquisition Tool* that allows the experts to quickly find new training images, so they can find new or changed objects with the object detection. For optimal accuracy, we used original images from the test drive video as training data. For fast training data search, we utilize the combined data. We integrated a checkbox area, Fig. 1A, where the experts can plot all data in an interactive line chart, Fig. 1B. The video data are the number of detected objects for each image with the currently trained object detection and with a confidence higher than a selected threshold ϕ. These checkboxes are created from each dimension dynamically, so Caarvida can handle changes in the log and video data (R-adapt). To see the corresponding images, the user can click on a data point in the line chart (Fig. 1B). Users can select the current image to store it automatically in the *Training Image Pool*. This approach helps domain experts quickly find training images. For example, if they want to detect stop signs, they can find images with stop signs faster if they focus on images where the speed of the car is zero. The user can further change the threshold value ϕ to find training data that contains detected objects with a low confidence.

(D) Label Tool. To support the domain experts in efficiently labeling training images for the object detection, we modified the existing tool *LabelImg* [19]. We also enabled access to the *Training Image Pool*, where domain experts can see which images are already labeled. Additionally, we integrated an automatic labeling process from the currently trained object detection method.

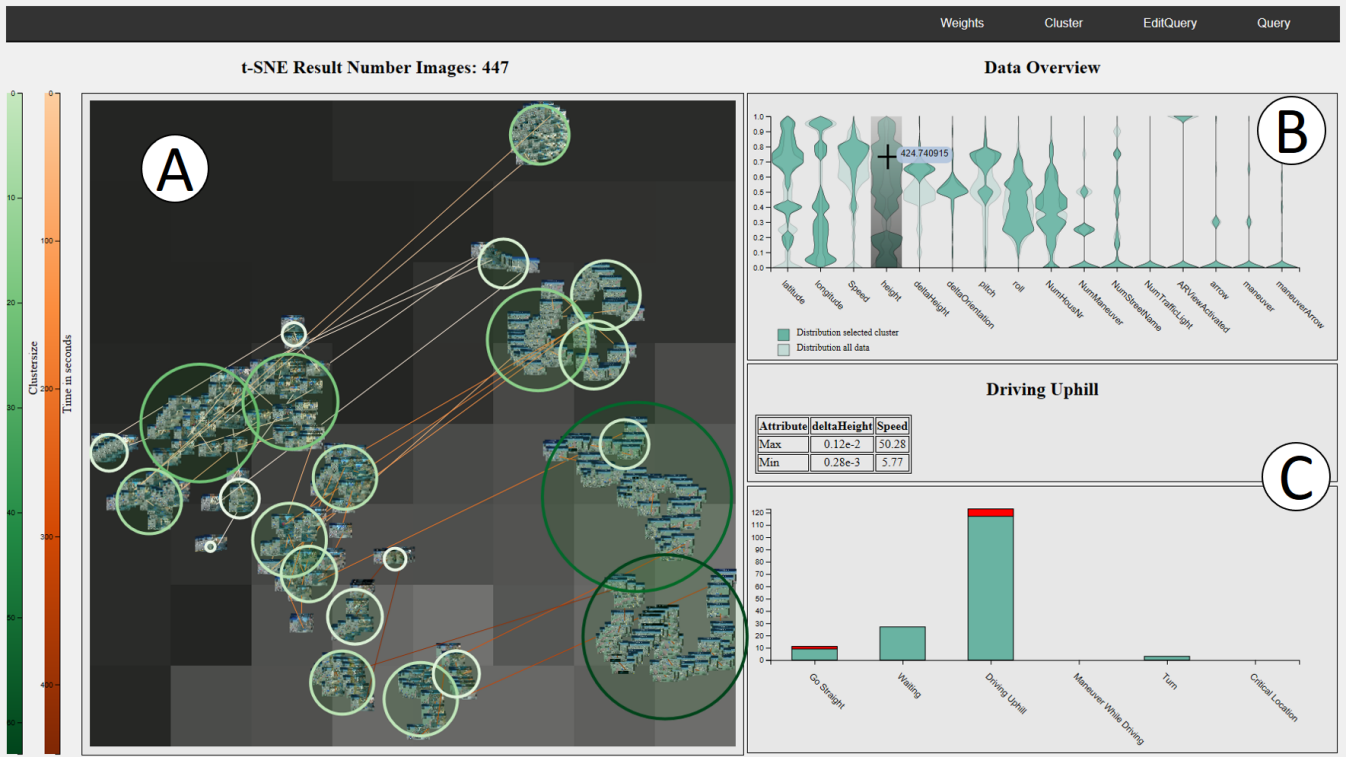

Figure 3: The Data Analysis Component with the dimension reduction result of the combined log and video data and a heatmap of a currently selected dimension (A). The data points are represented with miniature images of the video. The lines are between two images, indicate they were recorded in succession. The green circles are clusters which allow the user to select multiple images at once. (B) The interactive violin plot shows the distribution of each dimension of the data. (C) Top: The query table which shows the minimum and maximum values of the current query. Bottom: The result of all stored queries which show how many images of the current video meets the query requirements. The red portion of the bars show how many images have objects that are outlier depending on their position.

4.2 Data Analysis Component

In the *Data Analysis Component*, we developed an *Interactive t-SNE and Violin Plot* to generate an overview if the video has the necessary scenes to solve the task. The *t-SNE Metric Manipulation* enables the experts to cluster the task dependent data. The *Image View* supports the users to accomplish the task by analyzing multiple images simultaneously. Afterwards, the relevant data can be stored in a *Query* to find the task dependent images in other videos automatically.

Interactive t-SNE. To support the domain experts in determining whether a video contains task relevant scenes (R-overv), we need an approach that can visualize our high dimensional data. There are many techniques available, such as scatterplot matrices, parallel coordinates, or glyphs, each of those results in visual clutter with an increase in dimensions. Therefore, a dimension reduction method, like PCA, MDS, or t-SNE is appropriate. For our dimension reduction, we use t-SNE [5] because it considers the local structure of the data, which is important for our tasks, because similar errors are likely to be found in the local neighborhood of the data. We extend the t-SNE result with additional information to gather knowledge about temporal changes and the images. We split the video into keyframes and run t-SNE only on the data of the keyframes. Each data point is represented by a miniature image of its keyframe, Fig. 3A. Seeing these images allow the experts to recognize patterns or outliers, such as image clusters where the AR navigation system is turned off. We add lines connecting consecutively recorded images to get temporal information, such as whether the AR navigation system turned off at the end or the beginning of the video. To get more details of the data and images of a cluster, we added a selection technique that uses DBSCAN in the low dimensional space. We use the low dimensional space for performance reasons, so experts can select clusters interactively during t-SNE is optimizing.

Interactive Violin Plot. To get a deeper insight into the actual data, we must see the values of the data, such as the mean / minimum / maximum speed of the test drive or whether the car drove uphill. To answer these questions, we need to inspect the distribution of all data dimensions and the actual values. We therefore integrate an interactive violin plot, Fig. 3B. This plot shows the dimensions' distributions of the whole data as well as the selected cluster's data. Additionally, users can see the actual value of the distribution by hovering over it, as shown in the tooltip (Fig. 3B). We also include

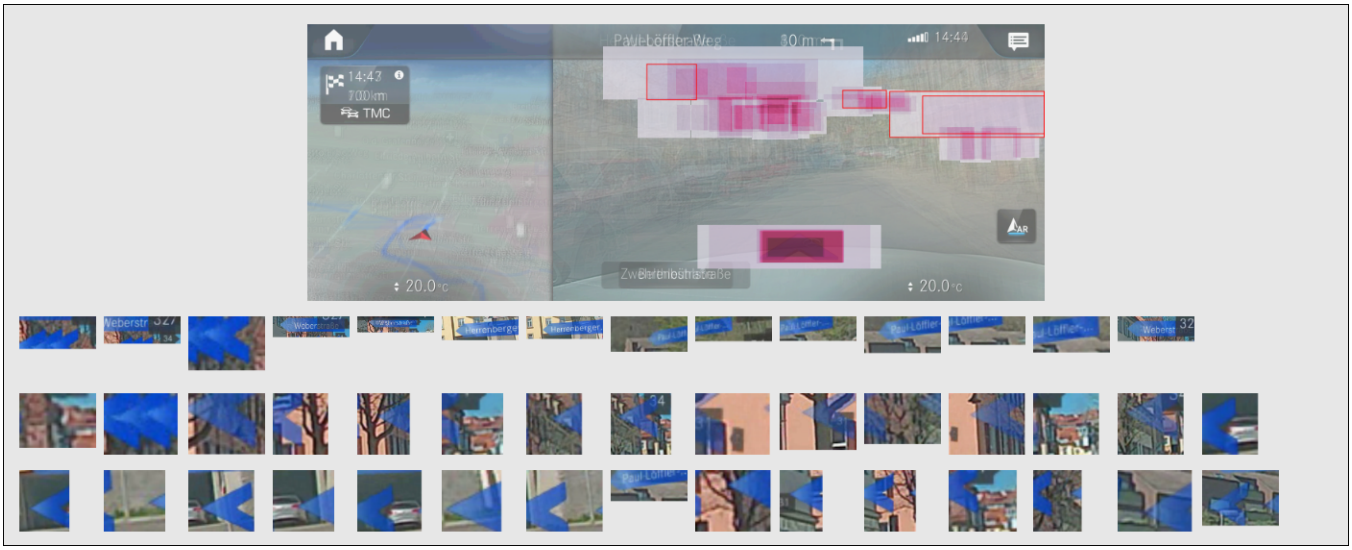

Figure 4: The image view of a selection of images. On top is the average image of the selection and a heatmap of the detected object. At the bottom are all detected object extracted in one list.

a heatmap in the t-SNE plot that appears after hovering over one dimension in the violin plot and displays information from the t-SNE plot, as shown in Fig. 3A. Here, the experts can see that the cluster in the bottom right consists of data with the highest height. This method is introduced by Stahnke et al. [13]. They focus on methods helping users to understand dimension reduction results better and faster. Another way to get information about the data of the t-SNE plot is to use intervals in the violin plot. The user can set an interval on one or multiple features. Afterwards, the data points in the t-SNE plot with the corresponding feature values in that interval, are drawn with a red border. Compared to the heatmap from Stahnke et al. this method has the benefit of showing the distribution of the data in the t-SNE plot with multiple dimensions instead of one.

t-SNE Metric Manipulation. After gathering an overview of the data, the experts must find the data that define the scene, that contains the task relevant images. To do so, they will collect them in one cluster. Data of t-SNE clusters are based on all dimension. If two images have the same speed and number of maneuver objects but all other dimension differ, it is likely that they are not in the same cluster. But experts may want to see all images with the same speed and number of maneuver objects, which would mean searching multiple clusters for these images. To cluster the images depending on the interest of the domain experts, we modify the t-SNE similarity calculation metric to get a dimension reduction result based on the weighted dimensions, without the influence of unnecessary dimensions (R-task). To control the influence of each dimension, we introduce a method:

We add a normalized weighting vector W that consists of separate weights w_i where $w_i \geq 0$ and $i \in [1, ..., N]$ and N is the number of dimensions. Afterwards, we change the Euclidean distance $d(x_i, x_j)$ from t-SNE to:

$$d'(x_i, x_j) = \|W(x_i - x_j)\|_2 \tag{1}$$

With this modification, the domain experts can cluster the images based on the task. If one dimension is not important for the current task, they can set its weight to zero. For example if they only care about the dimension "speed" they can set the weight of "latitude" to zero. If a dimension is necessary but other dimensions are more important, they can set the weights depending on their importance. With this approach, they can select a task relevant cluster, see its data in the violin plot and its images in the *Image View*, and check if this data includes the task relevant images.

Image View. T-error is about analyzing images and finding navigation objects that could confuse the driver. To do so, experts need to compare multiple images and identify possible differences between them. Within these images, they then need to understand where an object of interest lies, whether it is in the correct position in the context of the video, and whether it is readable. For example, if the domain experts have to find an image with a maneuver object too high above the street or they want to find navigation objects with mutated vowels in their text.

We developed an image view with an average image and a heatmap of the detected objects from a selection of images, Fig. 4. In the average image, the experts can recognize similar and different areas, such as street contours or multiple objects in the same position. To focus on task dependent images (R-time), we enable users to limit images to those with objects in which they are interested in by hovering and scrolling over the bounding boxes. Below the average image, we show extracted objects of the image selection, so users can check the object's readability. From the literature, we identify multiple ways to visually supporte this. For instance, we could do an image comparison like VAICo [3], but this does not work for images that are recorded from a moving camera. Another

idea would be to calculate "Eigenslides" via PCA [10] to identify similar regions in images. But PCA would take too much time to calculate for our images, so we adapted this idea for our purpose.

Querys. After the experts complete one task, they usually repeat it with other videos. To handle this more efficiently, we enable the user to create queries (R-time, R-repet). Here, the user stores the data defining the scenes necessary for the task to find them automatically the next time. Here, the experts use the relevant data from the selected cluster containing the task-relevant images checked in the *Image View*. For example, there are critical locations in some cities with complex street structures. With queries, experts can save the specific latitude and longitude of these locations to find them automatically in other videos. To create a query, users enter the data values they want to find in a table. They can enter the data manually or via intervals in the violin plot.

5 ANALYSIS OF TIME SAVINGS

We evaluated Caarvida's time reduction via a quantitative analysis and conducted two usage scenarios to demonstrate how Caarvida prepares data and defines a query to repeat a specific task automatically.

5.1 Time Reduction

To evaluate the time savings that Caarvida provides, we conducted a quantitative analysis inspired by GOMS [2], a classical evaluation approach based on user modeling. The main idea behind GOMS (simplified) is to break down higher-level goals into lower-level operators, which allows a completion time to be reasonable estimated. Adding up the times of the different operators needed gives an estimate of the overall time needed to reach a certain goal. We argue that in our case, this approach gives a good estimate of the potential time savings. A future quantitative lab study might be useful to further refine these results, as GOMS results are known to be imprecise.

As a first step, we break the high-level goals down into lower-level operations for both the baseline approach and our Caarvida approach. Then, we accurately define how long each operator takes. For the current practice, experts only have a common media player, so their only operation is clicking on the timeline to skip to different parts in the video. How often they repeat this operation depends on the task and video complexity. Because the amount of operation repeats for complex tasks is too varied, we focus on low-complexity tasks. Low task complexity means that the task should be accomplished by just skimming over the videos to notice only whether a navigation event occurs, not the details of it. To estimate the required time, we skimmed over three different test drive videos (total duration about 26 minutes and 40 seconds) with different complexities and measured the time. We need **232s** to skim over all three videos. To switch to a new video, we estimate **5s**.

To compare these values with Caarvida, we must perform a similar task. Here, we are looking for a specific crossroad in the same three videos at the same time. The time to accomplish the task in Caarvida is depends on whether there is already a query for this task or not. First, we assume there is a query, though we analyze the time without a query later. To complete the task with Caarvida, we perform three high level operations: (1) load data, requiring **30s**, depending on hardware performance, (2) click on the query result and draw average image (**15s**), (3) use the *Image View* by scrolling (**30s**), so in total we need **75s**. Based on these estimations, the domain experts would take about **232s** using their current practice, which is more than three times as long. The time reduction with Caarvida would increase even more with an increased number of videos to be analyzed.

5.2 Usage Scenario 1: Data Preparation

In this usage scenario, we demonstrate how Caarvida handles new features and enables experts to efficiently find training data. We focus on Task 4 and assume that the AR navigation system has a new feature that uses traffic light information. It is important to know where in the videos traffic lights appear in order to evaluate the quality of the new feature. Experts can use the *Training Data Acquisition Tool* to find images with traffic lights quickly. Here, they plot the speed data in the line chart to see when the car is stationary and analyze the corresponding images. It is likely that there are traffic lights in images when the car is stationary, because it could be waiting for a traffic light to change. Experts can then select these images for the training data. Fig. 1 shows this example.

After they collect multiple training images, they have to label them in the *Label Tool*. There they see which images are not labeled and can easily select them. These images are already analyzed by Yolo, so that the detected objects do not have to be labeled manually. The domain experts only correct these labels if necessary. After labeling the traffic lights, they can start to train Yolo. Afterwards, the experts can test the training results with another video. If the accuracy of traffic light detection is insufficient, they can find more training images by applying a threshold value to find traffic lights with a low confidence instead.

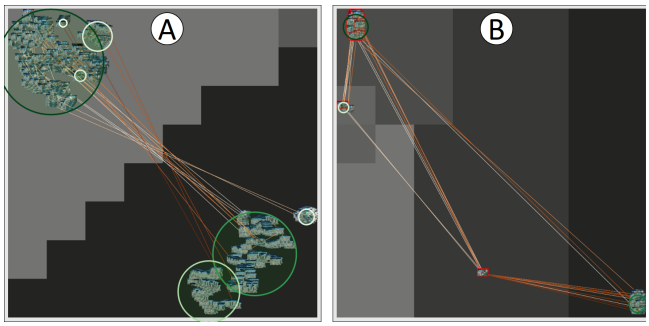

Figure 5: The t-SNE results of the usage scenario 2: (A) the result after weighting the t-SNE metric weight "ARViewActivated" to high and the others to low and its heatmap for the same attribute. (B) the result after "maneuver" weight to high and the others to low and its heatmap. Here, we can see that there are only images with 0,1,2 or 3 maneuver objects.

5.3 Usage Scenario 2: Query Creation

In this usage scenario, we show how users can find task-relevant images and create a query using the data that contains these images. Then, we calculate how much time is saved when this task is repeated with subsequent videos. We focus on Task 1 (T-error) where we want to find maneuver objects that are too high above the street during uphill driving. We use a test drive video with a duration of 7min and 37s.

To calculate how much time we need for this process, we use the same approach as above. For the current practice, we have to skim over the same video carefully to identify any errors. Here, we need **120s** to skim over the video and find the errors.

Using the workflow of Caarvida, we implement several different operations. The first high level operation is to load data, which takes **14s**. We then delete data that obviously cannot include the error. Our second high-level operation is to set the weight for "ARViewActivated" to high in order to remove images where the AR navigation view is turned off, Fig. 5A. This operation requires a drag and drop gesture to apply the weight, waiting for t-SNE optimization, hovering over the data where the AR view is turned off (which can be recognized via the images), and clicking delete. In total, we need **25s** for this operation. We do the same for "maneuver" to delete images without maneuver objects (**25s**), Fig. 5B. The third high level operation is to set "speed" and "deltaHigh" to high, find and click on the cluster with the images that could contain the errors by using the violin plots intervals, and changing the cluster parameters (**45s**). The final operation is to draw the average image (**2s**). We then scroll over the bounding boxes to check the position on the road and complete the task for this video (**5s**). Now, we see that the cluster contains the task-relevant images, Fig. 4, and create a query with the important data. We type in a name, draw intervals in the violin plot over the distribution of the latest selected cluster for the attributes "deltaHeight", "speed","numManeuver" and "ARViewActivated" and click on save (**17s**). In total it took **133s** for these steps.

Compared to the current practice, we see that it takes only a few seconds more to complete the task and create a query. To repeat the task for other videos, we only have to run the query and watch the videos in the *Image View*, which is much faster than the baseline practice, we showed in section 5.1.

6 CONCLUSION

In this design study, we developed a visual analytics tool called Caarvida that supports automotive engineers in their test drive video analysis workflow. The problem we adress in this study is reducing the time spent analyzing multiple test drive videos with navigation information annotations from an AR navigation system. Caarvida helps the engineers gather additional data from the video and handle data changes via an object detection method and a *Training Data Acquisition Tool*. Furthermore, we include visual interactive components such as interactive t-SNE projection and violin plots to provide a quick overview of the data and allow users to quickly find task-relevant images. We show that Caarvida significantly reduces the time to accomplish a task when a task is repeated for multiple videos.

Transfer-ability. Caarvida is also transferable to other data and tasks, such as analyzing soccer game videos players movement data. If we have players' x and y and the balls' position for each frame, the violin plot would show their distribution. Additionally, we could calculate the distance to the goals for each player and show them in the violin plot too. With this information, analysts could see if a team would play more offensively or defensively based on various ball positions. They could also analyze the position of a player while the opponents are attacking, in order to find errors the players make. They could then save the interesting situations in queries and find them in other videos automatically.

Limitations. Measuring a realistic and accurate accomplishing time is challenging because it depends on many factors. Such as the complexity of the tasks. There are some tasks where experts neet to watch the video very carefully, like whether identifing the fading of maneuver objects is appropriate (T-improv). But there are some less complex tasks like checking the maneuver object's position on a specific complex crossroad (T-error). The accomplishing time also depends on the amount and complexity of the videos. A complex video consists of less waiting and fewer sequences where the AR navigation system is turned off. Due to these factors and their high variance of occurrence, it is challenging to evaluate a general task-accomplishing time. Also, Caarvida is limited in that it does not reduce the task-accomplishing time when it is used for short videos separately (less than two minutes), unless queries are used. Caarvida should rather be used for longer videos or repeated tasks.

Future Work. We plan to evaluate Caarvida in a longitudinal study to gather further information regarding how we can improve Caarvida and to learn about its value in other environments.

ACKNOWLEDGMENTS

This work was supported by the FFG ICT of the future program via the ViSciPub project (no. 867378).

REFERENCES

[1] Rita Borgo, Min Chen, Ben Daubney, Edward Grundy, Gunther Heidemann, Benjamin Höferlin, Markus Höferlin, Heike Leitte, Daniel Weiskopf, and Xianghua Xie. 2012. State of the art report on video-based graphics and video visualization. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 2450–2477.

[2] Stuart K Card. 2018. *The psychology of human-computer interaction.* Crc Press.

[3] Steven R Gomez, Radu Jianu, Caroline Ziemkiewicz, Hua Guo, and David Laidlaw. 2012. Different strokes for different folks: visual presentation design between disciplines. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2411–2420.

[4] Philip A Legg, David HS Chung, Matthew L Parry, Rhodri Bown, Mark W Jones, Iwan W Griffiths, and Min Chen. 2013. Transformation of an uncertain video search pipeline to a sketch-based visual analytics loop. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2109–2118.

[5] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[6] Gérard Medioni, Isaac Cohen, François Brémond, Somboon Hongeng, and Ramakant Nevatia. 2001. Event detection and analysis from video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23, 8 (2001), 873–889.

[7] Himani S Parekh, Darshak G Thakore, and Udesang K Jaliya. 2014. A survey on object detection and tracking methods. *International Journal of Innovative Research in Computer and Communication Engineering* 2, 2 (2014), 2970–2978.

[8] Matthew L Parry, Philip A Legg, David HS Chung, Iwan W Griffiths, and Min Chen. 2011. Hierarchical event selection for video storyboards with a case study on snooker video visualization. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (2011), 1747–1756.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 779–788.

[10] Johanna Schmidt, M Eduard Gröller, and Stefan Bruckner. 2013. VAICo: Visual analysis for image comparison. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2090–2099.

[11] Michael Sedlmair, Petra Isenberg, Dominikus Baur, Michael Mauerer, Christian Pigorsch, and Andreas Butz. 2011. Cardiogram: visual analytics for automotive engineers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1727–1736.

[12] Michael Sedlmair, Miriah Meyer, and Tamara Munzner. 2012. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2431–2440.

[13] Julian Stahnke, Marian Dörk, Boris Müller, and Andreas Thom. 2015. Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2015), 629–638.

[14] Manuel Stein, Halldor Janetzko, Andreas Lamprecht, Thorsten Breitkreutz, Philipp Zimmermann, Bastian Goldlücke, Tobias Schreck, Gennady Andrienko, Michael Grossniklaus, and Daniel A Keim. 2017. Bring it to the pitch: Combining video and movement data to enhance team sport analysis. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 13–22.

[15] Jeffrey A Stevens. 2007. Visualization of Complex Automotive Data: A Tutorial. *IEEE Computer Graphics and Applications* 27, 6 (2007), 80–86.

[16] Yoshimasa Takahashi, Naoko Nitta, and Noboru Babaguchi. 2005. Video summarization for large sports video archives. In *2005 IEEE International Conference on Multimedia and Expo*. IEEE, 1170–1173.

[17] Marcus Tonnis, Rudi Lindl, Leonhard Walchshausl, and Gudrun Klinker. 2007. Visualization of Spatial Sensor Data in the Context of Automotive Environment Perception Systems. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, 115–124.

[18] Trieu. 2017. Darkflow. https://github.com/thtrieu/darkflow

[19] Tzutalin. 2015. LabelImg. https://github.com/tzutalin/labelImg

[20] Yingcai Wu, Xiao Xie, Jiachen Wang, Dazhen Deng, Hongye Liang, Hui Zhang, Shoubin Cheng, and Wei Chen. 2018. Forvizor: Visualizing spatio-temporal team formations in soccer. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2018), 65–75.