Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Tool-Supported Data Collection for Experiments to Subjectively Assess Vision Videos

Bachelorarbeit

im Studiengang Informatik

von

Vignesh Arulmani Sankaranarayanan

Prüfer: Prof. Dr. Kurt Schneider
Zweitprüfer: Dr. Jil Klünder
Betreuer: M.Sc. Oliver Karras

Hannover, 23.09.2019

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 23.09.2019

Vignesh Arulmani Sankaranarayanan

Zusammenfassung

Die Anforderungsermittlung ist eine wichtige Prozess der Anforderungserhebung, da die herausgekitzelten Anforderungen eigentlich in gut festgestellten Anforderungen umgewandelt werden, die dann weiter als Bausteine für ein effizientes Softwareprojekt dienen. Im Bereich von Elicitationstechniken funktionieren Videos als einer der eindeutigste und deskriptive Wege zur Kommunikation vom Problem, der Lösung und der Ergebnisse einer Prozess alles in kurzer Zeit zu den Stakeholdern. Wenn Methoden zur Verbesserung der Qualitätsaspekten der solchen Vision Videos entwickelt werden, wird es auch die Performanzrate der Anforderungserhebungsprozess des geförderten Projekt vielfach erhöhen.

Der in dieser Arbeit entwickelte Werkzeug, *Feedback Recorder*, zeigt eine Implementation einer detaillierten subjektiven Qualitätsbewertung, die zur Erfassung der potenziellen Videoqualitätsdaten dient, die sonst verliert gehen können. Grundsätzlich speichert das Programm die Qualitätsbewertungen, die während des Laufen des Videos eingegeben wird. Stattdessen normalerweise nur eine solche Bewertung zum Ende des Videos gespeichert wird. Im Zusatz dazu könnte das Werkzeug die gespeicherten Daten unterschiedlicher Teilnehmer eines Versuchs in grafischer Form darstellen. Dies könnte dann auch zum sofortigen Vergleich und der Erkennung der Schwachstellen in den Videos führt. Die Evaluierungsprozess dieser Arbeit handelt sich um das Fördern der Usabilitätsaspekten des Werkzeuges mithilfe von relevanten Metriken.

Abstract

Tool-Supported Data Collection for Experiments to Subjectively Assess Vision Videos

Elicitation of requirements is a crucial process in Requirements Engineering as it is what is chiseled into final requirements that form the building blocks of an efficient software product. Out of the many techniques used to elicit requirements, videos stand out from the rest due to their unique and descriptive way of conveying the problem, solution and the results of a process to the involved stakeholders in a short period of time. When methods are developed to enhance the quality aspects of such Vision Videos, their increased performance would help accelerate the requirements engineering process of the promoted project.

The software tool developed in this thesis, known as *Feedback Recorder*, presents an implementation of a detailed subjective quality assessment that could be used to collect potentially useful video quality data that might otherwise get lost. The program primarily helps in storing the quality scores of data throughout the length of a video instead of just an objective score at the end of the video. Additionally the tool can also represent results of different participants of an experiments in a graphical form that could help in instant comparison and identification of weak spots in the videos. The evaluation process of this thesis involves promoting the usability aspects of the tool with the measurement of relevant metrics.

Contents

Chapter 1

Introduction

1.1 Motivation

In this day and age, the increasing number of problems in our daily lives has led to the alarming need for reliable software-based solutions. The application of engineering principles to software, i.e. Software Engineering, plays a major role in the procedural development of such software in a quick and efficient manner. A crucial branch of Software Engineering that acts as a backbone to the features of a software is Requirements Engineering. It helps in identifying, eliciting and validating the functional and qualitative aspects of the software. Requirements are elicited with a greater vision in mind, that is specific to the particular problem at hand. They specify the functions and characteristics that the future system is supposed to fulfill in order to be transformed into a complete final product. Although the requirements are generally textually extracted, another tool that can be used to vividly visualize the overall process before the developmental stage are Vision Videos, as understood from Karras et al. [13]. A Vision Video illustrates a real world scenario that addresses the existing problem, a visionary solution to it and the impact of that particular solution, according to the definition cited from Mergenthaler [16]. The stakeholders responsible for the designing and handling of Vision Videos could be requirements engineers or even entire development teams. Thus it can be noticed that the Vision Videos help the stakeholders in the Elicitation and Validation phases of the Requirements Engineering process according to Pham et al. [19]. Although videos are a more expressive way of documentation and help in improving the understability of a project, it should be noted they are comparatively expensive to produce [12]. If the quality of the videos is moreover not assured through evaluation, then the results of the Requirements Engineering process could be faulty or prone to errors, as is the case generally with software engineering according to Boehm et al. [4].

Objective evaluation depends on deriving results from fixed video quality

metrics, but the results might still not define what a good video is, as the outcome may vary when observed subjectively. That is, by collecting the reviews of different subjects and testing on the similarities and differences between them. The human communication factor brings with it an effective way of requirements gathering and analysis, according to Holtzblatt et al. [8]. Subjective evaluation helps in introducing this missing factor into the equation. For example, the emotional quotient that the viewer experiences when watching a video could only be rated subjectively. And so, commonalities in spikes between participants and reoccuring tendencies may discover a new outcome from a different dimension with respect to the quality of the video. Seshadrinathan et al. [22] have included such a notable setup as a part of their experiment in which the participants subjectively assess the quality scores of a video after watching the whole video. In terms of video quality it is not entirely enoughto make an evaluation entirely on the basis of one final subjective assessment at the end of the video. The results do portray how different individuals rated the quality of the video, but a different technique is required to identify the locations in the video where drastic changes in opinion might have taken place. There could hence be a possible loss of additional information, crucial to the accurate assessment of the video quality, that originates from not having the access to these temporal changes of quality scores. Hence an enhanced solution is required in order to repair the inadequate nature of the information.

1.2 Goals

One possible solution that could overcome the problem regarding the lack of potentially useful information, is performing a detailed subjective assessment throughout the entirety of the video. The aggregation of the cumulative results could help pinpoint the exact spots where a quality aspect is well implemented or inversely when more correction is required. This could in turn end up adding more support to the relevance of the procured data. Since it would primarily be costly and furthermore inconvenient to the participant, to devise a manual approach to implement the above procedure, this thesis suggests the development and usage of a tool instead, i.e. a computer application named *Feedback Recorder*, to achieve the same. The application should uphold the quality aspects from the ISO-9126 Software Quality Model [23] [14], that are most relevant to the problem at hand as stated by Jamwal et al. [9], namely *Functionality* and *Usability* here. Hence the metrics that are to be tested in the quality assessment portion of the paper have to be based on these particular aspects. The description below shows how the chosen quality aspects are represented within the working of the application.

The two primary goals that are going to be elaborated and expanded upon at further sections of this thesis are as follows:

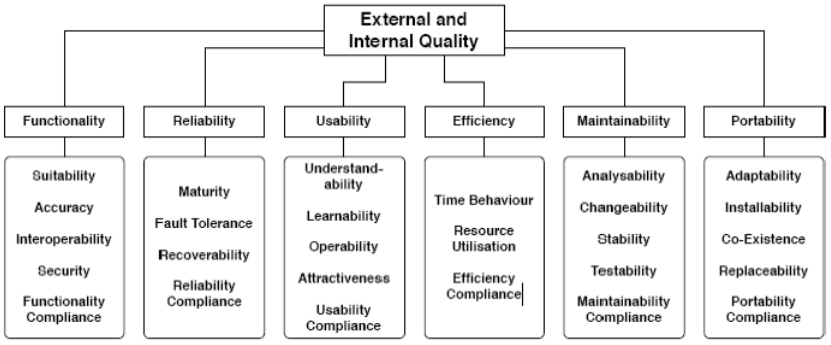


Figure 1.1: ISO 9126 Software Quality Model with Attributes [10]

- **G1**: *To highlight the functionality of the tool.*
  The application should be designed in such a way that during the recording process the to-be-analyzed Vision Video should be visible on the application window and when played, the viewer, i.e. the participant, should get to input their rating with respect to the metric being tested (let's just say for the sake of understanding, the *image quality* of the video), and should even be able to change the rating as the video progresses. The entered quality scores should be stored in a readable format along with other additional assessment information that is obtained from the participant. Tendencies of rise or dips in quality scores from the different participants could then be compared, analysed and evaluated from the collected data, hence assisting the detailed subjective assessment process.

- **G2**: *To assess and improve the usability of the tool.*
  The input method should be easily and intuitively operable and should come with different types of input, hence making itself considerably attractive to the user. The value distribution in the scale used in the input method should be understandable, at least through the use of labels to clear doubts around possibly ambiguous information.

The ISO 9126 Software Quality Model in Fig. 1.1 describes Functionality as a functional attribute and Reliability, Usability, Efficiency, Maintanability and Portability as other non-functional qualitative attributes. Since this thesis concentrates on highlighting the functionality of the application (from main goal G1) and assessing the Usability aspects of the tool (based on goal G2), the vision would be narrowed down to only these two aspects described in the ISO 9126 Model. As of this application, once the primary goals are claimed to be met, the metrics corresponding to the goals (mainly the goal G2, as quality factors such as usability of an application are tested from the perspective of subjects) are to be broken down and tested by the

experimenter with the help of an efficient quality assessment model to imply the implementation of the overarching goals that encompass the metrics.

1.3 Structure of Thesis

The content of the thesis is divided and categorized into six chapters which are structured in the following manner. The first chapter provides an introduction into the motivation behind the topic of the thesis and the basic goals that are to be met by the end of the thesis. In a nutshell it deals with the development and testing of a tool that collects data in experiments to subjectively assess Vision Videos in depth. In the second chapter the fundamental concepts that are required to understand subject-specific terms, definitions and processes in the later parts of the thesis are explained in detail. The individual processes of Requirements Engineering, the definition and uses of Vision Videos are furthermore handled over here. In the third chapter we get an insight into the processes that were used in regular cycles to extract and refine the needed requirements through the known *Elicitation* and *Negotiation* phases of Requirements Engineering. The functional and non-functional, i.e. predominantly qualitative, requirements conceptualized from the primary goals with the help of the aforementioned processes are listed here. The fourth chapter is initiated with a description of the core concepts derived from the previously extracted requirements. After which the profound functioning, the underlying software principles and the examples surrounding the *Feedback Recorder* tool are discussed thoroughly for the most part. The section *Related Works* gives a brief view into other scientific papers that deal with experiments or topics similar to the ones in here. The fifth chapter opens with the details regarding the quality assessment model used to derive the relevant metrics from the known requirements and hypotheses. The remainder of the chapter revolves around the experiment conducted to qualitatively assess the tool and the evaluation of the results procured from the survey. The final chapter briefly summarizes all the discussed topics and provides a concise outlook for future topics that could be extended from this paper, thus marking the end of the thesis.

Chapter 2

Fundamentals

This section presents fundamental concepts that form the groundwork for the research done in this thesis. The thesis presents a software-oriented methodology that works on enhancing the use of vision videos. Therefore it is essential to know the basics of requirements engineering, which is used to develop the feedback recording tool and the idea behind vision videos.

2.1 Requirements Engineering

Definition 1 (Requirements Engineering)**.** *Requirements engineering is concerned with identifying, modeling, communicating and documenting the requirements for a system, and the contexts in which the system will be used (cited from Paetsch et al. [18]).*

The field of Requirements Engineering helps provide context and groundwork to the carrying-out of a software engineering procedure. Tools such as processes, techniques and scenerios help in strengthening this groundwork, as observed from Nuseibeh et al. [17].

The reference model from Fig. 2.1 compactly spans out the constituents of the Requirements Engineering methodology.

2.1.1 Requirements Analysis

The System Analysis phase of Requirements Engineering acts as a step-by-step process gathering data from the concerned sources, which are then extracted and refined into concrete requirements. The Requirements Analysis consists of five steps: *Elicitation*, *Interpretation*, *Negotiation*,*Documentation* and *Validation/Verification*, which are explained in detail below.

- **Elicitation:**
  The process involves extracting the characteristics and functionalities of the yet-to-be-developed software from the clients and stakeholders

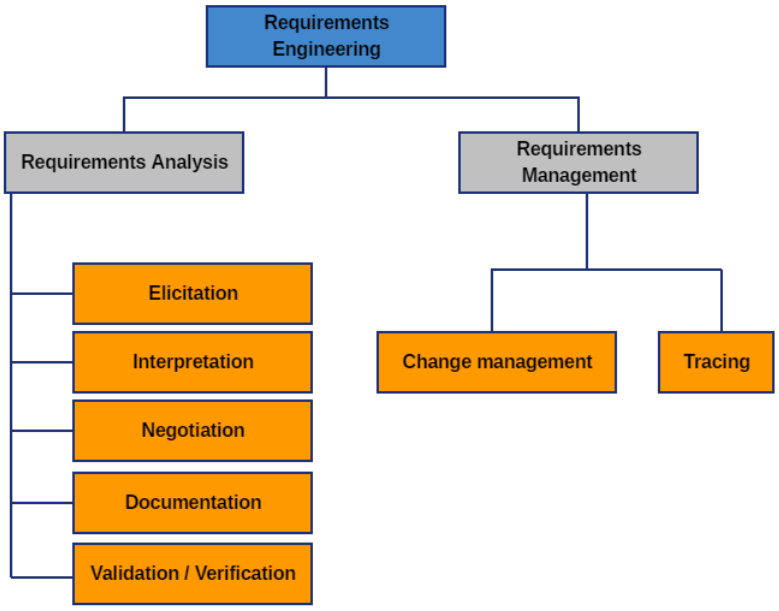


Figure 2.1: Requirements Engineering Reference Model [5]

as a first step. The extraction of information is performed roughly with the help of standard communication processes such as Interviews, textual techniques like Question Forms, Concept Videos, etc. This step is crucial in the Requirements Engineering procedure as it is responsible for acquiring the raw information, which acts as a basement for the fine-tuning of requirements in the following phases.

- **Interpretation:**
  The extracted raw notes are put forward for contentual interpretation, through which points displaying similar patterns are categorized, structured and sorted to get a more defined outline. The underlying principle here is to separate the relevant information out so that it can be distinguished from the unnecessary information.

- **Negotiation:**
  Negotiation involves the handling of the intermediate-level requirements. The processes in Negotiation concentrate on identifying dependencies, inconsistencies in the requirements and deciding whether or not to resolve them.

- **Documentation:**
  The refined requirements need to be documented in order to gain a fixed tangible state, but at the same time they should also be change-

able in response to possible future updates. It is necessary to explain the requirements with relation to the involved variables additionally in the documentation along with the requirement definitions.

- **Validation and Verification:**

  The cycles of validation and verification are used to ensure the quality of the documented requirements and to minimize, detect and correct errors at such an earlier phase so that they do not persist, as stated by Maalem et al. [15]. The objective of requirements validation is to check contentwise if the documented requirements match with the description the system. Whereas requirements verification formally checks if newly constructed designs that are to be implemented, match with the previously documented requirements.

2.1.2 Requirements Management

The Requirements Management phase of Requirements Engineering deals with the issue of handling possible changes to the validated requirements smoothly so that it does not disrupt the whole system. The steps required for this management process are *Change Management* and *Tracing*.

- **Change Management:**
  The usage of Version control systems and Management techniques to handle versions and changes in the requirements engineering process constitutes the idea of Change Management.

- **Tracing:**
  Tracing promotes the documentation of the process in such a manner that it shows the relation to the product. It requires keeping records of the sources for the system requirements and also the kinds of changes that happened to the system post-tracing.

2.2 Vision Videos

As it is observed from Fig. 2.1, the position of the usage of Videos (*Video taping* and *Video conversation*) are high on the graph relative to most other requirements elicitation techniques.

Also Fricker et al. [7] suggests using Videos in Workshops as it is a feasible technique to efficiently conduct indirect communication between the developer and the client.

Definition 2 (Requirements Engineering)**.** *A Vision Video illustrates a real world scenario that addresses the existing problem, a visionary solution to it and the impact of that particular solution. (cited from Mergenthaler [16])*

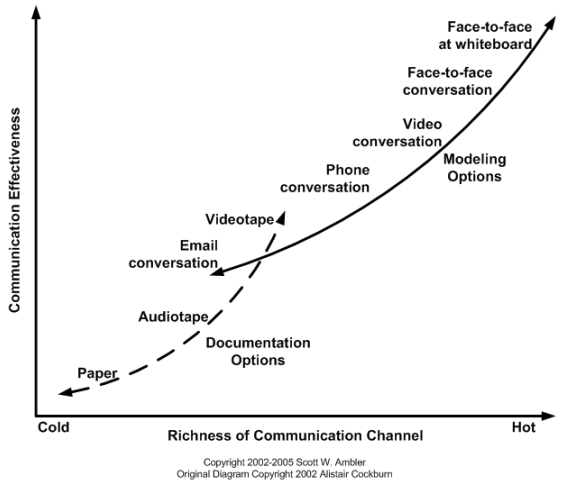


Figure 2.2: Modes of communication [1]

Vision Videos can be watched together with the stakeholder and can account for instant validation, according to [19]. Vision Videos are generally easy to understand and can hence help infuse understandability and expressiveness into the communication process between the stakeholder and the developer, as explained further in the thesis.

Chapter 3

Requirements

This chapter deals with the process of eliciting specifications and converting them through the Requirements Engineering process (mentioned in the previous chapter) into meaningful documented requirements for the further working of the application.

3.1 Design Process

The design process chosen for this project is a custom mix of traditional and agile methodologies. The process involves compressing the phases of the traditional Requirements Engineering into a weekly iteration frame. The iterations are built upon in an incremental manner until the product is ready to be dispatched. In the meeting at the start of the week, the *design* of the application and results in user stories being sketched. The *elicited* user stories are refined and prioritized based on a series of *interpretations* and *negotiations*. Requirements are drafted and *documented* from the user stories and are used to *implement* the code during the week. The functioning of the implemented code is discussed and *analyzed* at the beginning of the following week's meeting. The validation of the code marks the end of an interation. Leftover story cards are integrated to the list of story cards created in the new iteration's design process. *The system development life cycle is the overall process of developing, implementing, and retiring information systems through a multistep process from initiation, analysis, design, implementation, and maintenance to disposal*, cited from [20]. This cycle applies the *Design-Implement-Analyze (D-I-A)* routine which is in turn a model adapted from the mentioned *Software Development Life Cycle (SDLC)*. It is also noteworthy that the described semi-agile process shows usage of eXtreme Programming (XP) practices such as *Short Releases* (weekly), *Continuous Integration* and the usage of *Coding Standards*.

3.2 Scope Requirements

Scope Requirements for the overall application are linked to making informed decisions about the external factors of the project, such as the operating system or the programming language used. In a way the scope requirements act as a form of constraint.

- **[R10]**: *The application should be able to run on all operating system platforms.*
- **[R11]**: *The application is programmed using JavaFX.*
- **[R12]**: *Serialization and Deserialization of stored text is done using* ***FX-Gson*** *format.*

3.3 Functional Requirements

The functional requirements defined below are primarily based on the goal G1, i.e. to highlight and strengthen the impact of specified functionalities to increase the usefulness of the tool.

3.3.1 General Requirements

The General Requirements that contain the overall essense of the idea behind the application are listed below.

- **[R20]**: *Experiment entries can be added, viewed and deleted.*
- **[R21]**: *Video path and input method type are to be defined in an Experiment entry.*
- **[R22]**: *Participant entries can be added, viewed and deleted.*
- **[R23]**: *Either Slider or Radio Buttons can be selected as the input method.*
- **[R24]**: *Data that is input through a predefined mouse-based action while the video is playing can be stored for being saved later to a file in readable format.*
- **[R25]**: *Data collected from all participants for a specific experiment can be displayed in graphical form.*
- **[R26]**: *The Data Graph should be linked to the video through an action.*
- **[R27]**: *Data collected from all participants for a specific experiment can be exported to a file in a readable format.*

- **[R28]**: *The application can be closed from the Main Menu and all screens can be redirected to the Main Menu.*

3.3.2 Individual Requirements

The functional requirements of the application are described below for each and every individual pane and window.

Main Window:

- **[R30]**: *Saved Experiment Entries are displayed on the Experiments List.*
- **[R31]**: *Saved Participant Entries are displayed on the Participants List.*
- **[R32]**: *New Experiment or Participant Entries can be added upon action.*
- **[R33]**: *Information about existing Experiment and Participant Entries can be viewed upon action.*
- **[R34]**: *Existing Experiment and Participant Entries can be deleted upon action.*
- **[R35]**: *Information about existing Participant Entries belonging to a particular Experiment can be exported to a file in readable format upon action.*

Experiment Window:

- **[R40]**: *Experimenter can choose the Experiment video.*
- **[R41]**: *Experimenter can choose Slider or Radio Buttons as an input method.*
- **[R42]**: *Experimenter can customize minimum and maximum label names for all input types.*
- **[R43]**: *Experimenter can set minimum and maximum bounds for the slider.*
- **[R44]**: *Experimenter can customize the number of radio buttons and their respective label names.*
- **[R45]**: *Experimenter can add new fields to the demographics form.*
- **[R46]**: *Experimenter can activate or disable the additional assessment form.*

- **[R47]**: *The selected settings are saved and can be viewed later.*

User Info Window:

- **[R50]**: *User ID and additional demographic information of the participant can be entered and stored.*

Recorder Window:

- **[R60]**: *Participant can change Slider values (or else Radio Button values) while watching the video.*
- **[R61]**: *The values selected in the Slider (or in the Radio Buttons) are saved along with their corresponding time-points in the video.*
- **[R62]**: *Video can also be paused (and played again) midway.*
- **[R63]**: *Video can be rewinded to an earlier point in the video and all scores from that point onwards would then be renewed.*
- **[R64]**: *Volume of the video is adjustable.*
- **[R65]**: *Data should be saved to the file only after the entire video is watched.*

Question Form Window:

- **[R70]**: *The additional assessment information entered by the participant can be saved*

Data Graph Window:

- **[R80]**: *Clicking on a node of the data graph sets the time slider of the video to the corresponding time.*
- **[R81]**: *Screenshots can be taken at any point in the video and can then be viewed immediately.*
- **[R82]**: *The recorded data from multiple Participant Entries of a particular Experiment can also be displayed in a single graph.*
- **[R83]**: *Mean, Median and Standard Deviation graphs can also be displayed for multiple Participant Entries of an particular Experiment.*

Info Viewer Window:

- **[R90]**: *Saved demographic information and assessment scores of the participant can be displayed*

Menu Bar:

- **[R100]**: *Information about the product can be displayed.*
- **[R101]**: *Application can be closed at any time while active.*

3.4 Qualitative Requirements

The qualitative requirements defined below are defined around the idea of usability from G2.

- **[R110]**: *Recorded and Exported data are stored in .csv format*
- **[R111]**: *Video height has been fixed to 250.*
- **[R112]**: *Number of Radio Buttons are restricted to 10.*
- **[R113]**: *Application is set to full screen.*

Chapter 4

Concepts

In Chapter 3 we saw how the initial process of eliciting specifications began in our project and led to their transformation into concrete documented requirements. In this chapter we shall see how the core conceptual topics are derived from the requirements, how the implementation of the Tool (*Feedback Recorder*) took place and what kind of programming principles were used in the process. Furthermore, the works related to this thesis shall be looked into in detail at the end of the chapter.

4.1 Topics

The documented requirements from the previous chapter can be grouped into sets of topics based on similarity in fields. When each of the filtered topics constitutes a different testable area that is relevant to the basic concept behind the tool, we then get the Core Topics, as follows:-

- **T1: Customizable Input**
  This topic is based on the requirements [R23], [R41], [R42], [R43], [R44], [R45] and [R46]. The topic deals with the customizable nature of the available input methods, namely *Slider* and *Radio Buttons* (refer to Fig. A.2 and Fig. A.3). Customization revolves around the different options that an user can use to achieve the same goal, i.e. but using different means. In this case, customization primarily helps the person playing the role of the experimenter in creating a custom experiment as per the associated requirement needs. Hence it comes down to which input method fits the experiment better according to the experimenter.

- **T2: Continuous Data Collection**
  This topic is based on the requirements [R24] and [R61]. It suggests the storage of the quality scores (mapped with respective video time) entered by the participants using the input method, continually throughout the Vision Video while it is playing (refer to Fig. A.6 and

Fig. A.7). This continuous nature of the collected quality assessment data is what that helps the experimenter determine the high and low points of the video. And also whether or not the found gaps are chosen to be acknowledged and refined. It is also to be noted that according to [R65] the collected data can be saved to a file only at the end of the recording session, that is when the video ends.

- **T3: Simultaneous Visualization of Distributed Data**
  This topic is based on the requirements [R25], [R26], [R82] and [R83]. The continual quality scores of all participants of a particular experiment can displayed in graphical form. The Data Graph is set right below the corresponding video. During the post-experiment phase, the experimenter can analyse the stored results by going through the graph. Upon mouse-clicking on a graph node, the time slider of the video above changes its value to that of the corresponding time. This simultaneous visualization of data helps the experimenter observe, compare and evaluate the results and can further use this additional data to refine the video, as mentioned in the topic T2. Additional modes have been built into the graph that calculates the **Mean**, **Median** or **Mean (+/-) Standard Deviation** values from all data values for each second and represents the resulting graphs as the output of these modes (refer to Fig. A.11, Fig. A.12, Fig. A.13 and Fig. A.14). These additional features exist so that the experimenter could use them to identify otherwise unnoticeable trends immediately with ease and could choose to take them into account if interested.

- **T4: Immersive Recording Experience**
  This topic is based on the requirement [R62]. The implemented requirement would offer the option to pause the video during the recording process, adjust the rating while the video is on pause, and then play it again. This gives the participant the choice to step back when their sense of immersion is disturbed (for example:- when they need to ask the experimenter a question in the midst of a video) and then get it intact again. The immersiveness of the recording experience deals with whether the participants were fully able to get into the recording process while watching the video. An important metric that would suggest otherwise is whether the participants felt it to be distracting to review the video simultaneously while watching the video. This factor hence turns out to be an important criteria in determining the relevance and integrity of the collected results. Moreover, it impacts greatly the usability aspect of the tool.

- **T5: Reversible Nature**
  This topic is based on the requirements [R28], [R63]. The first specification provides the participant with the option to **Rewind** a

video to a previous video in the middle of the recording process, if the participant wishes to change their entered feedback from a particular onwards. The second one ensures that an user can **Quit** the application from the main window and that a **Back** or a **Cancel** option is provided in every other window to get back to the main window. These requirements are to be implemented to emphasize the efficient reversibility of the application, which is a suitable quality aspect under which the program could be tested.

The topics T2 and T3 are important pillars that uphold the *Functionality* of the tool with respect to the experimenter, i.e. related to the main goal G1. Whereas T1, T4 and T5 are responsible for enhancing the *Usability* aspects of the tool with respect to the participant, i.e. they relate to the main goal G2. These topics would be tested and evaluated further in the following chapter.

4.2 The Tool: Feedback Recorder

The tool, titled *Feedback Recorder*, provides an application platform for the Experimenter to create custom experiments that involve subjectively assessing the quality aspects of Vision Videos. This section explains the design and implementation processes of the tool in detail.

4.2.1 Technical Overview

The *Feedback Recorder* tool is essentially a desktop application developed using *JavaFX*. Since JavaFX has been removed by Oracle from *Java Development Kit 11* onwards, it might be necessary to install JavaFX from an open-source alternative like *OpenJFX* (link: `https://openjfx.io/`). The IDE used for the development is *Eclipse Mars (4.5.2)*. *Gson* (or *Google Gson*) is an open-source Java library used for the serialization and deserialization of Java objects to (and from) JSON; link: `https://github.com/google/gson`. Our tool, *Feedback Recorder* uses the aforementioned *gson* along with *fx-gson* (refer to the JSON examples), which is a set of type adapters for Google Gson to make serialization of JavaFX properties more natural; link: `https://github.com/joffrey-bion/fx-gson`. It is highly essential that the **JavaFX**, **Gson** and **fx-gson** packages are installed in the system beforehand (packages are to be installed the aforementioned open-source links, if not already present) for *Feedback Recorder* to work. Furthermore the FXML files, that contain the design portions of most windows, have been designed with the help of the open source application *SceneBuilder* (that is backed by *Gluon*); link: `https://gluonhq.com/products/scene-builder/`. The simple drag-and-drop functionality offered

by *SceneBuilder* helped saving time in comparison to writing the desing code programmatically.

4.2.2 Implementation

This section explains the technical details behind the implementation of the *Feedback Recorder* tool. The entire process of the users interacting with the system can be vividly described with the help of the Use-Case-Diagram here. The *Experimenter* and the *Participant* are the involved actors, whereas all data generated and recorded from the application is stored in the *Experiments* folder (acts as database, refer to the package structure in Fig. A.16). Taking *Screenshots* of specific points in the video and *Exporting* all recorded data into a CSV file are the optional features mention in the diagram. The remaining mandatory steps of the procedure are mentioned below with respect to the individual scenes displayed in the application.

- **Main Menu:**

  When the application is opened, the *Main Window* (refer to Fig. A.1) is the first screen to appear. This window is mainly responsible for displaying the lists of experiments and the participants for each experiment. Further more, processes such as creating an experiment entry or initiating the recording process can be conducted here. Clicking on the add experiment (+) button changes the screen to the Experiment Window. Once a new experiment has been added, the experiment name would be shown in the Experiments List View on the left side of the screen. Right-clicking on any of the individual experiments opens a context menu in which the following options are displayed: *View*, *Show All Data*, *Export* and *Delete*. *View* opens the Experiment Window in view mode and displays the stored settings of that particular experiment. *Show All Data* opens the Data Graph Window where the stored data of all participants of that experiment are graphically displayed. Selecting *Export* packs the stored experiment settings and the recorded data of all the participants into a single CSV file (*export.csv*). Pressing *Delete* deletes that particular Experiment's folder and all its contents. On the other hand, the add participant (+) button is disabled by default, and is enabled only when a particular Experiment Entry is selected. Once the add participant (+) button is enabled, clicking it changes the screen to the User Info Window and initiates the Recording process. Once the recording session has successfully completed, the new participant is then added and that participant's ID would be shown in the Participants List View on the right side of the screen. Right-clicking on any of the individual participants opens a context menu in which the following options are displayed: *View*, *Show Data* and *Delete*. *View* opens the Info

Viewer Window in view mode and displays the stored settings of that particular experiment. *Show Data* opens the Data Graph Window where the stored data of that particular participant is graphically displayed. Pressing *Delete* deletes that particular Participant's folder and all its contents. Clicking on the *Quit* button at the bottom of the screen closes the application.

- **Experiment Screen:**

  The screen opens into the *Experiment Window* (refer to Fig. A.2 and Fig. A.3), that is normally used by the Experimenter for creating a new experiment. The *Experiment Name* field asks for an unique experiment name to be entered. The Vision Video used for the entire experiment must be selected using a file chooser in the *Selected video* field. Then either *Slider* or *Radio Buttons* can selected as the appropriate *Input method* in the corresponding combobox. The *Settings* section in the bottom part of the scene are linked to the input method selected in the combobox, hence they alternate accordingly when the value is changed. Additional name fields can be added to *Demographic Settings*, but the unique *ID* field remains fixed and cannot be omitted. The additional assessment form can be activated or disabled in the *Assessment Settings*. If *Slider* is selected as the input method, then the following attributes are needed to be filled:- *Min Label* and *Max Label* denote the label names on the left and right most ends of the slider, i.e. they describe the attribute names on either extreme ends. *Min Value* and *Max Value* denote the minimum and maximum bounds of values on the slider. There is additionally a *Preview* Label that shows the currently selected value on the slider to give the user an idea as to how it functions. If *Radio Buttons* is selected instead as the input method, then the following attributes are needed to be filled:- *Min Label* and *Max Label* are defined just like for Slider. Radio Buttons settings additionally offer the option to input the number of radio buttons in *No. of buttons* and the entered value triggers the creation of the exact number of labels. The names entered in the radio button *Labels* are how the Likert values would be displayed. *Min Value* and *Max Value* denote the minimum and maximum bounds of values on the slider. There is additionally a *Preview* Label that shows the currently selected value on the slider to give the user an idea as to how it functions. Upon clicking the *Next* button the entered values are parsed into (JSON with the help of the FxGson builder and other self-defined private classes such as *ExperimentController.Experiment*, *ExperimentController.Settings*, etc.) stored in the *settings.json* and *extra_ settings.json* files as seen in the package structure. In view mode, only the labels in the *Settings* section and the *Experiment Name* are editable, whereas the data of all other attributes (extracted from the

JSON files) can only be viewed.

- **Recorder:**

  As mentioned earlier, the recording process is triggered by clicking the add participant (+) button. The screen first shifts to the *User Info Window* (refer to Fig. A.8), which contains the *ID* field and all the additional demographic fields extracted from *demographics.json*. It is recommended to fill in a four-digit number into the *ID* field, which is later used to represent that particular participant within the application, hence promoting anonymity.

  Upon clicking *Next*, the screen changes to the *Recorder Window* (see Fig. A.6 and Fig. A.7). The recorder window consists of a Video Player (displaying the selected video) on the top of the screen and the selected input method (Slider or Radio Buttons) on the bottom of the screen. The Video Player used in the application is an adaptation of a MediaPlayback model described in *Oracle Docs*; link: `https://docs.oracle.com/javase/8/javafx/media-tutorial/playercontrol.htm`. The player has been modified such that, it is linked to the input method below. Certain disable factors and the overall layout of the pane have also been modified appropriately. In addition to it *Screenshot* (disabled here, but enabled in the *Data Analyzer* screen) and *Rewind* functionalities (enable here, but disabled in *Data Analyzer*) have also been added to the custom *MediaController.java* class.

  The video is paused in the beginning and the *Next* button is initially disabled, so that the entire video is watched before proceeding to the next step. Once the video is played, the recording process begins and the participant gets to take control of the mouse to input their ratings. The input values are centered by default, but the changes in the values (*value*) over time are stored in a Map variable as key-value pairs along with the time (*key*). As it can be intuitively figured, the value change in a Slider is done through sliding the key and in the case of radio buttons, the change is done by distinctly selecting another radio button in the group. The participant can pause the video any time, change the rating (quality score) comfortably and again continue playing the video using *Play/Pause* button. If the *Rewind* button is clicked, then a new dialog opens, showing a slider with 0.0 seconds as the minimum value and the current time in seconds as the maximum value. The user gets to choose the time to which the video is to be rewinded and from which the stored is to be deleted. At the end of the video, the previously disabled *Next* button is then enabled and upon clicking it, the recorded information is saved to be stored into a CSV file at the very end (provided the following assessment process does not get cancelled).

The screen then moves onto the *Question Form Window* (see Fig. A.9). The participant has to answer a list of subjective questions. THEY require the user to enter overall ratings for a predefined list of video quality attributes (rated from 2 to -2 based on the Likert Scale, where *2* is *Very High*, *1* is *High*, *0* is *Neutral*, *-1* is *Low* and *-2* is *Very Low*) with descriptions as seen from Fig. A.9 and Fig. A.10. The form also includes a general question about the *overall video quality* which provides only two choices, i.e. either *good* or *bad*). When all values are selected and *Next* is clicked, the recorded map data is finally stored into a CSV file (*data.csv*) and the currently filled assessment form data gets parsed and stored into a JSON file (*assessment.json*) in the Participant's Folder. Thus marking the end of the entire recording process.

The participant is now visible in the list in the main window. Upon double-clicking it or selecting the *View* context menu option, the *Info Viewer Window* is opened. Here the stored *Demographics* and *Assessment* data of the participant are extracted from the existing files and are displayed in a tab pane for viewing.

- **Data Analyzer:**

  The *Data Graph Window* can be displayed in two different ways: (i) The Data Graph for a single participant (refer to Fig. A.4), that is triggered by right-clicking on a participant and selecting the *Show Data* option. (ii) And the Data Graph for multiple participants of an experiment (refer to Fig. A.5), that is triggered by right-clicking on an experiment and selecting the *Show All Data* option. The Show All Data option is primarily the same as the Show Data option, except for the fact that the recorded continual data of all participants of an experiment are converted into different graphs and displayed together into a single *Line Chart* (that is a type of Graph display in JavaFX).

  The visual description and functionality of the Data Graph Window fulfills all the aspects mentioned in T3 accordingly. The *Data displayed* combo box is directly linked to the graphs below, and the selected mode here is *All* by default (refer to Fig. A.11). When the selection of the combo box changes, then the graph changes accordingly to display the corresponding data with respect to whether the new mode is *Mean*, *Median* or *Mean (+/-) Standard Deviation* (Fig. A.12, Fig. A.13 and Fig. A.14).

  Upon clicking the nodes on any of the graphs, the user can access the point in video corresponding to that particular time key of that node. In the Video Player above, the *Rewind* button is disabled and the *Screenshot* button is enabled. A screenshot can be taken at any point in the video and it is immediately saved to the system and displayed

in a list of screenshots for that particular participant (if *Show Data* chosen) or experiment (if *Show All Data* chosen instead) to the right side of the Video Player. All the stored screenshots are displayed with the help of *Pagination Control*.

- **Menu Bar:** The application has a *BorderPane* layout and the menu bar is pinned to the Top part of it. There are two Menu Items on the bar: *File* and *Help*. *File* contains the *Close* option which is responsible for closing the application at any point. *Help* contains the *About* option which opens a Dialog Box, where information about the project and the university can be viewed.

Hence, it could be derived that through the implementation of the tool's functionality, the main goal main G1 has been achieved. The graphical results acquired from the tool in the following chapter would accentuate this statement further.

4.3 Related Works

This thesis relies heavily on ideas and concepts surrounding *Vision Videos* and the subjective assessment tools in relation to videos. Therefore the similar works of several other individuals mentioned below, greatly helped in narrowing down the direction in which the research of this thesis had to proceed to achieve fruition.

Schneider et al. [21] performed an experiment with 20 subjects divided into 2 groups, in which they were shown videos shot using subjectiveand third-person perspective and their preferences were measured. The experiment was conducted with a Goal (based on the GQM model) to *refine* vision videos into more detailed and concrete scenarios in an affordable manner. The results showed that although there were no significant difference between the subjects' preference for one of the two perspectives, there were differences between the two groups/sets in the degree to which the participants preferred one style of videos (i.e. the subjective perspective style). They highlight therefore how the inclusion of ratings, which is a subjective assessment tool, in such an experiment shows difference in the grade of rating between participant results, that are similar on the surface (i.e. here the *liking of the video*).

Karras et al. [12] developed a software tool (*ReqVidA - Requirements Video Analyzer*) that highlights relevant sections of a video and attaching notes. The idea was to combine the textual and video-based elicitation techniques into a single tool to provide easy and fast access to relevant information. In the evaluation phase of their project, an experimental setup was conducted, in which 12 participants used the tool as scribes. Using the GQM model, the *time* and *quality* of elicited requirements were

measured, so that requirements of higher-quality could be detected. Our tool (*Feedback Recorder*) follows a similar procedure with respect to the tool development and the evaluation methodology. But our tool differs by focusing on improving the quality of videos based on the offered detailed subjective assessment instead.

In the article written by Seshadrinathan et al. [22] there exists a portion of an experiment that involves subjectively assessing videos with the help of a subjective study interface. The participants were shown videos using the interface and were prompted to quality scores for the overall video at the end of the video. The assessment option uses *Slider* as an option with marked label ratings (based on Likert scale) below the slider. Whereas the concept of the *Feedback Recorder* goes a step further and provides the option for participants to provide their quality scores with the help of Likert-based input methods throughout the entire video in addition to an overall video quality rating at the end.

The thesis presented here is partly an extension of an idea obtained from the outlook (German: *Ausblick*) section of the master thesis of Karras [11]. His work describes the initial development of the previously mentioned tool *ReqVidA* and the evaluation done using it. Some of the possible extensions mentioned for in his outlook, that were taken and worked further on in our project, are as follows:

- *Improvement of the weak spots in the video identified in the heuristic evaluation conducted.*
- *A more intensive evaluation of the results from the Workshop participants to improve the prototype.*
- *A more comprehensive integration of videos into the whole Requirements Engineering process or even Software-development process.*

Feedback Recorder gets closer to solving the above issues through the implementation *continuous* and *detailed* subjective assessment of video qualities. The data collected (exported) at the end of an experiment provides an insight into how each participant has assessed certain attributes and how their results compare to those of other participants from the same experiment. The analysis performed by an experimenter on these results act as an intensive evaluation technique which helps in identifying weak spots in the video for further correction and improvement. This highlights the integration of the videos into Requirements Engineering processes such as Validation, Verification or even Change Management. Hence the position of the Feedback Recorder in the Requirements Engineering process and the Quality Management processes of the Vision Videos has been justifiably strengthened.

Chapter 5

Evaluation

The chapter that describes the Evaluation process, where the implementation of the tool defined in the previous chapter is to be highlighted and the usability aspects of *Feedback Recorder* are to be thoroughly tested using an efficient experiment setup and execution.

5.1 Concepts behind the Experiment

Fundamental hypotheses need to be derived from the previously defined core topics, with the help of a quality assessment model, before proceeding into the survey process.

5.1.1 Goal-Question-Metric (GQM)

Contains information regarding the GQM Model.

The *Goal Question Metric* (GQM) model, explained by Basili et al. [3], forms the basis of the qualitative evaluation process. According to Caldiera et al. [6], the GQM approach involves specifying goals depending on the purpose that one is trying to achieve. The next step suggests breaking down the goals into the questions, which when answered fulfill the connected goal. The questions are in turn broken down into necessary metrics, that could be tested through an experimental procedure. After performing the concerned experiment, the measured results of the metrics come together to solve the linked questions.

The custom GQM sheets (refer to Fig. 5.1, Fig. 5.2 and Fig. 5.3) derived from the previously defined topics concretized from the following Goals:-

- *Assess the usability of the customizable input method from the stakeholder's viewpoint*
- *Assess the immersiveness of the recording process from the stakeholder's viewpoint*

- *Assess the usability of the tool's reversible nature from the stakeholder's viewpoint*

Goal	Purpose Issue Object (process) Viewpoint	Assess the usability of the customizable input method from the stakeholder's viewpoint
Question		Which is the better input method?
Metrics		• Number of participants who selected Slider • Number of participants who selected Radio Buttons

Figure 5.1: Screenshot of GQM sheet derived from topic T1

Goal	Purpose Issue Object (process) Viewpoint	Assess the immersiveness of the recording process from the stakeholder's viewpoint
Question		Was the recording process distracting?
Metrics		• *Distraction factor*, d.f. - based on Likert scale - possible values (1: Strongly Disagree/ 2: Disagree/ 3: Neutral/ 4: Agree/ 5: Strongly Agree) • Number of participants distributed based on their selection of d.f.

Figure 5.2: Screenshot of GQM sheet derived from topic T4

Goal	Purpose Issue Object (process) Viewpoint	Assess the usability of the tool's reversible nature from the stakeholder's viewpoint
Question		How often is the video rewinded during the recording process?
Metrics		• Number of participants who rewinded at least once • Number of participants who did not rewind at all • Total number of rewinds • Total number of participants

Figure 5.3: Screenshot of GQM sheet derived from topic T5

5.1.2 Hypotheses

The following hypotheses are constructed within the bounds of the participants interviewed in our particular experiment. The IF conditions

constitute indepedendent variables and the THEN conditions constitute the dependent variables.

The single Logical hypothesis that is postulated from the diagram is as follows:-

- $H_{1,1}$: (derived from Fig. 5.1) *IF the number of participants who selected* ***Slider*** *is not equal to the number of participants who selected* ***Radio Buttons*** *THEN the input method selected by the greater number of participants has the higher usability rating.*

The initial null hypotheses presupposed are as follows:-

- $H_{0,1}$: (derived from Fig. 5.1) *IF the number of participants who selected* ***Slider*** *is equal to the number of participants who selected* ***Radio Buttons*** *THEN the usability of the customizable input methods remains undecided.*

- $H_{0,2}$: (derived from Fig. 5.2) *IF more than half of the participants* ***Strongly Agree*** *with the statement THEN the recording process is claimed to be distracting to an extent and hence not immersive.*

- $H_{0,3}$: (derived from Fig. 5.2) *IF the mean of the distraction factors selected by all participants is greater than 3 THEN the recording process is claimed to be distracting to an extent and hence not immersive.*

- $H_{0,4}$: (derived from Fig. 5.2) *IF* ***Neutral*** *has been selected as distraction factor by all participants THEN the immersiveness of the recording process remains undecided.*

- $H_{0,5}$: (derived from Fig. 5.3) *IF the number of participants who rewinded at least once is equal to zero THEN the usability of the tool's reversible function remains undecided.*
  It can be noticed that the *number of participants who did not rewind at all* is exactly equal to the *number of participants who rewinded at least once* subtracted from the *total number of participants.*

- $H_{0,6}$: (derived from Fig. 5.3) *IF the total number of rewinds is equal to zero THEN the usability of the tool's reversible function remains undecided.*

5.2 Survey

This section defines the framework of the conducted survey and how the subject pool was chosen and how their participation took place.

5.2.1 Test subjects

A total of 8 students took part in the survey. It was observed that 7 students study at *Leibniz Univerität Hannover* whereas 1 student studies at *Fachhochschule Hannover*. And 6 of them are Computer Science (German: *Informatik*) students whereas 2 are Electrical Engineering (German: *Elektrotechnik*) students. Also 3 of the subjects are Masters students, whereas the remaining 5 are Bachelors students. Furthermore it could be observed from the demographic information (see Fig. A.21) that all students have passed the course *Software-Technik* (German for *Software-Engineering*), whereas 3 have passed *Software-Qualität* (German for *Software Quality*) and 4 of the students have passed *Software-Projekt* (German for *Software Project*). Additionally all students have given an affirmative response when asked if they had significant experience in programming. So it could be observed from the demographics that all students have a basic understanding of software engineering and programming concepts. They could hence be considered as a valid test pool for our survey.

5.2.2 Description of a Session

Each subject is shown two Vision Videos (*BioFeedback* and *OpticalEcho2*), where the input method (i.e. *Slider* or *Radio Buttons*) in the two videos are different from one another. Since there are two different types of videos and two different types of input methods, we get four possible combinations of experimental setups: *A*, *B*, *C* and *D*. Furthermore an *Evaluation Category*, i.e. a custom variable chosen for our case, is defined here as a code that consists of two alphabets, where the first alphabet denotes which video is to be first and similarly the second alphabet denotes which video is to viewed second. Considering both the videos have to be different and their different input methods should not be the same too, as mentioned earlier, we again get only four possible matchings of Evaluation Categories: *AB*, *BC*, *CA* and *DC*. Hence, also taking the above points into account, the mapping of the subjects (i.e. only with their self-chosen 4-digit *IDs*) to their respective *Evaluation Categories* can be seen from Fig. A.20. With this the survey setup has been established.

Here is an exemplary session as to how the survey was conducted. The session begins providing the subject with the *Experiment Overview* form (see Fig. A.22), that explains the ground idea behind the survey and the steps of the experiment. Additionally the approximate time to be taken (15-20 minutes) and the right of the participant to retract their participation anytime during the experiment without the need for reason have also been added to it. As a next step, the *Consent Form* (see Fig. A.23) is given to be filled, provided the participant is interested in taking part. The consent form ensures that the participant is physically able and willing to participate and

also explains the associated data privacy policy to them. The participant enters their name and a four digit ID chosen by them (that is going to be used in the experiment instead of their name) and signs the form.

After the above procedures, the experiment actually begins. The ID and the Evaluation Category of the participant is entered in the Experiment corresponding to the first alphabet of the Evaluation Category. Once the Recorder Window is opened, the participant is explained about the usage of the input method, meaning of the assessment labels (was *Bad Quality* and *Good Quality* respectively) and are made aware of the *Pause* and *Rewind* buttons. Then once the recording process starts, the subject rate the video continually while it is playing. Simultaneously as an experimenter I was noting the number of times the rating value changed (i.e. by observing overall drags in Slider and overall shifts in Radio Buttons) and the number of times the Rewind button used, in the form of Tallies in the *Additional Survey* (see Fig. A.25). After the video was watched the subject was followingly provided with the electronic assessment form (see *Question Form Window* from Fig. A.9) to be filled. At the end of it, the evaluation for the first video (from the first experiment) was stored. This process was repeated once again with the second experiment (derived from the second alphabet of the Evaluation Category), after which the survey results are completely processed, stored and exported.

As an optional additional step, an analysis of the recorded data was performed with collaboration with the participant. That is, the recorded continual quality scores provided by the participant were opened in a graphical form (refer to Fig. A.4) for a more intensive analysis of the data. The participant was asked to select video points on the graph that were particularly interesting to them, whether for a good or a bad reason, and screenshots were taken at those points. This optional step could be suggested to requirement engineers to elicit assessments of higher quality and precision in real-life scenarios. Thus, the sample session successfully comes to an end.

5.3 Analysis

5.3.1 Results from the survey

The results of the conducted survey that has been performed using the aforementioned setup, are presented here.

- ***Additional Survey*** **results**: The results here (refer to Fig. A.26) would further help in solidifying or disproving the initial hypotheses.
- ***Electronic Question Form*** **results**:

  The subjects assessed the *overall video quality* and also other important video quality aspects in the following order: *Image quality*,

Slider/RB?	Recording Process Disturbing?	#(REWIND)	#(VALUE CHANGES)
RB	Strongly Disagree	0	8 (A), 7 (B)
Slider	Disagree	4	8 (C), 7 (D)
Slider	Strongly Disagree	1	6 (B), 2 (A)
Slider	Strongly Disagree	0	6 (D), 8 (C)
Slider	Strongly Disagree	1	2 (A), 5 (B)
Slider	Disagree	0	5 (C), 10 (D)
Slider	Disagree	0	6 (B), 2 (A)
Slider	Neutral	0	2 (D), 1 (C)

Figure 5.4: Survey Results - *Additional Survey*

Sound quality, *Focus*, *Plot*, *Prior knowledge*, *Clarity*, *Essence*, *Clutter*, *Completeness*, *Pleasure*, *Intention*, *Sense of responsibility*, *Support* and *Stability* (refer to Fig. A.27, Fig. A.28, Fig. A.29 and Fig. A.30). Although the collected assessment data is more of an overall assessment taken at the end of the video, it surely gives a more detailed data set for the experimenter to work with than the just *overall video quality*, i.e. only whether the video is good or bad. When experiment groups with the same videos are compared, it could be observed that the order in which the videos were shown did not lead to a significant or noticeable difference in the *overall video quality* values.

- ***Data Graph* results**: The functionalities with respect to continual data collection (T2) and the simultaneous illustration of multiple graphs (T3) can be evidently seen from the data graph results. The Ups and downs between the graphs could be noted by the experimenter and improved upon. (refer to Fig. A.31, Fig. A.32, Fig. A.33 and Fig. A.34).

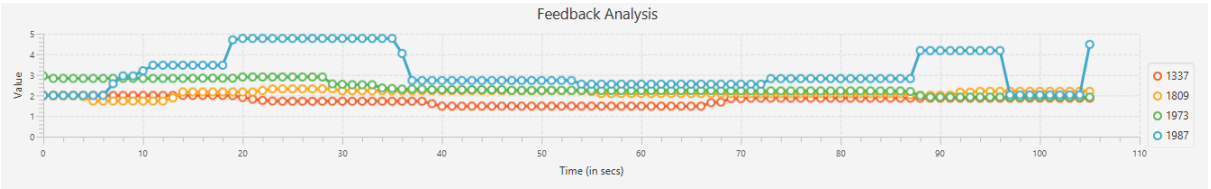


Figure 5.5: Survey Results - *Data Graph of* ***A***

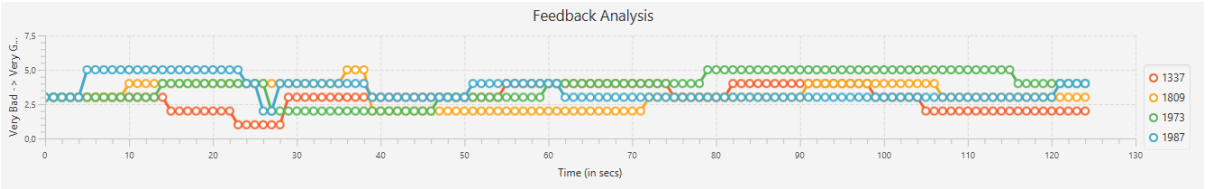


Figure 5.6: Survey Results - *Data Graph of* ***B***

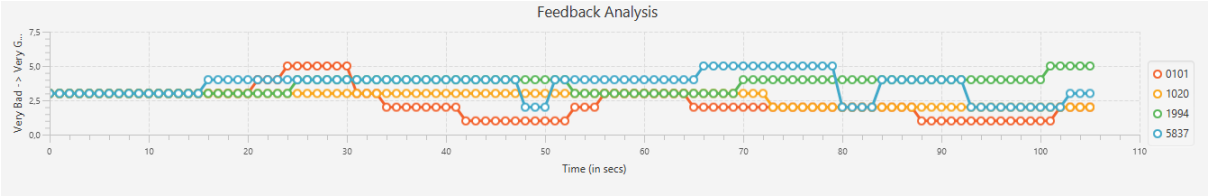


Figure 5.7: Survey Results - *Data Graph of* ***C***

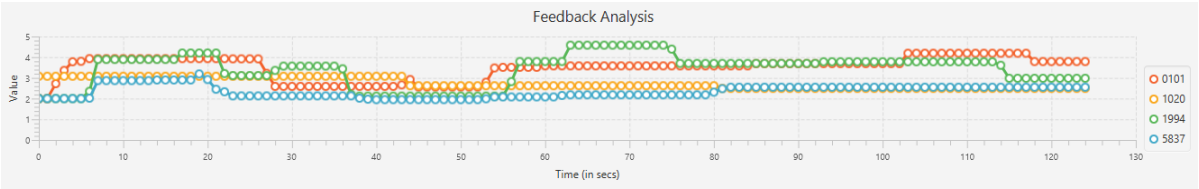


Figure 5.8: Survey Results - *Data Graph of* ***D***

Hence the comprehensive additional survey and data graph results obtained using the *Feedback Recorder* highlights the functionality of the tool in an understandable and verifiable manner. Finally the stakeholders get to decide how and when is implementation of the tool sufficient. So taking all the various factors as to how the implementation could be influenced, it could be accepted that the functionality of the tool has been adequately implemented.

5.3.2 Subjective Assessment

This portion deals extrapolating arguments from the results to support or counter previously defined hypotheses. Proving them to be right or not determines the kind of statement that they would make about the usability of the tool.

- It can seen from the graph (refer to Fig. 5.9) from the results that 7 subjects opted for Slider and only 1 subject for Radio Buttons. As 7 is greater than 1, it can be stated the Slider was preferred as the better input method by the tested pool.

  Hence the logical hypothesis $H_{1,1}$ has been proven from the above argument to be empirically true. On the other side the null hypothesis $H_{0,1}$ has been disproved.
  An alternate hypothesis, based on the derived empirical hypothesis, can be formulated that ***It is to be observed from the results, that Slider has been been preferred as a better input method than Radio Buttons***.

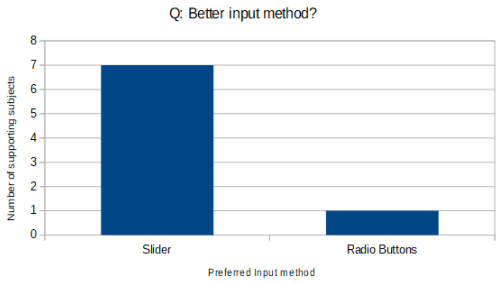


Figure 5.9: Survey Results - *Better input method?*

- The following arguments can be deduced from the graph (refer to Fig. 5.10).

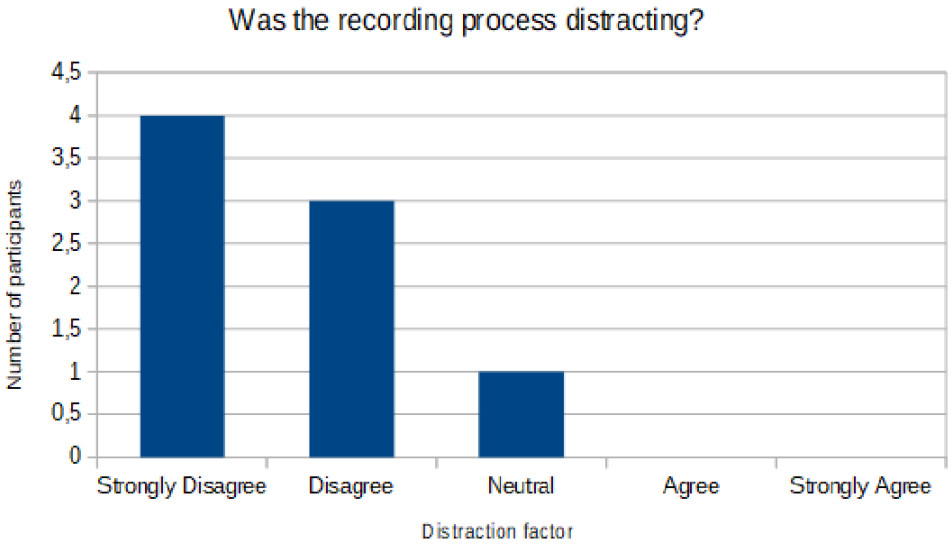


Figure 5.10: Survey Results - *Was the recording process distracting?*

(i) The number of subjects who opted for Strongly Agree is equal to 0, which is obviously less than 4, i.e. the half of the total number of subjects, 8.
(ii) The mean of the distraction values of all participants is **1.75** from the following calculation, i.e. where the product of the distraction factor and the respective number of subjects who opted it, divided by the total number of subjects, gives the following:

$$((4 * 1 + 3 * 2 + 1 * 3)/8) = 1.75$$

and clearly 1.75 is less than 3.
(iii) *Neutral* has been selected as a distraction factor only by 1

participant and not by all 8. Hence, from the arguments (i), (ii) and (iii) the following null-hypotheses $H_{0,2}$, $H_{0,3}$ and $H_{0,4}$ are disproved.
An alternate hypothesis can be formulated that ***It is to be observed from the results, that the recording process was not significantly distracting***.

- The *number of subjects who rewinded the video at least once* is equal to 3 (which is clearly more than 0). The *total number of rewinds* is equal to 6 (which is again more than 0), as observed from the additional survey results (see Fig. A.26). Hence the null hypotheses $H_{0,5}$ and $H_{0,6}$ are disproved by the above arguments.
  An alternate hypothesis can be formulated that **It is to be observed from the results, that the usability of the tool's reversible nature has been detected to a small degree, that is definitely not zero**.

The significance of null hypotheses are undermined, according to Anderson et al. [2], in which it is suggested to counter null hypotheses with multiple alternate hypotheses than a single alternate solution. But, since the scopes of the participant pool and the tested metrics are relatively small, providing an efficient alternate solution, that is strongly derived from results-based arguments, for every null hypothesis is sufficient.

The inductive derivation of the above alternate hypotheses is shown to enhance the usability of the *Feedback Recorder* tool, hence achieving the main goal G2.

Chapter 6

Conclusion

The final chapter concisely summarizes the most important themes dealt within this thesis. It finally brings the thesis to an end with a short outlook into how the features of the tool could be further improved in future projects related to this work.

6.1 Summary

Vision Videos play a major role in enhancing the Elicitation and Validation roles of the promoted projects. They are capable of providing clear, detailed and well-constructed visualization of to-be-implemented concepts. The video prototypes hence help in conveying the idea behind the ongoing project to stakeholders in a convenient manner. Since Vision Videos themselves are a software tool, they are also prone to descension in quality. So it is highly important to discover new methods and procedures to identify errors or inconveniences and correct them in order to uphold and manage the quality of the video. The purpose of this thesis was to develop a software tool that a more intensive and detailed subjective assessment of videos compared to existing solutions. Initially the necessary specifications were elicited, documented and improved into solid requirements upon in a series of interactions with the advisor. The consistent implementation of the requirements over the short period of time helped develop the now fully-functional *Feedback Recorder*. When the tool is used, the detailed subjective assessment of a selected vision video is performed in such a manner, that the measured data (quality scores) is collected continually throughout the video till its end. That is, data on the assessed quality aspect is collected every second while the video is being watched. The tool also provides the option to compare the recorded data of multiple participants of a particular experiment textually (*exported data file*) and graphically (*data graphs*) simultaneously. The requirements engineer could easily detect common dips at certain spots in the video and the reason behind it could be guessed from past experiences

or elicited from the participants during the experiment or a workshop. The goal-question-metric approach helped in narrowing down the relevant fields based on the purpose of the project - in this case assessing the usability of the tool. The breaking down of large goals into smaller comprehensible metrics, helped identify the appropriate tests needed to support the purpose of the whole endeavor. The results from the conducted survey highlighted the importance of the tool's functionality by showing how the individual core topics come together to give a detailed quality assessment in the case studies. But the main outtake from the results was finding out that the usability of the tool has been assessed positively under many criteria by the interviewed subjects. Hence the tool helps the stakeholders identify the current quality state of a vision video by presenting through a detailed assessment process. It leaves the remaining decision-making process of how sufficient the quality of the acquired information is, in the hands of the stakeholder. So with proper supervision and competent testers, *Feedback Recorder* could lead to massive improvements in the quality of the videos and in turn the requirements engineering process of the (product promoted in the video) in a considerably short span.

6.2 Outlook

A survey pool of 8 people is a relatively small amount to perform such an experiment on and was done so due to lack of time (specifically after developing the main application). And it could always help to perform the same or a similar experiment in a grander scale and check as to what the results then imply. But within the realms of this particular experiment, it was ensured that maximum amount of useful information could be extracted from the limited number of approached subjects, to assess the quality situation of the presented videos. Future experiments could also be tested with more videos, but it is noted that the permutations of the order in which the videos are to be shown also increases, hence increasing the overall effort put in the process.

The requirement with respect to developing the screenshot functionality was initially sketched with the intention to also store user comments at the captured locations in the videos. But due to time constraint, only a primitive version of the screenshot functionality could be implemented. The quality assessment concept behind *Feedback Recorder* could thus be integrated with applications that work on other aspects of the videos, such as applications that provide efficient elicitation techniques while watching the video. Such integration of similar tools could help requirement engineers to perform actions related to different fields of requirements engineering all in one place, and hence accelerates the software engineering process too.

Furthermore other kinds of input methods could be designed and

implemented, so as to increase the customizability and variety of the tool. For example, color graded boxes could help in getting the idea of the rating across to the an experiment subject in a intuitively manner and could hence possibly enhance the quality of the recorded scores too.

Appendix A

Appendix

A.1 *Feedback Recorder* Screenshots

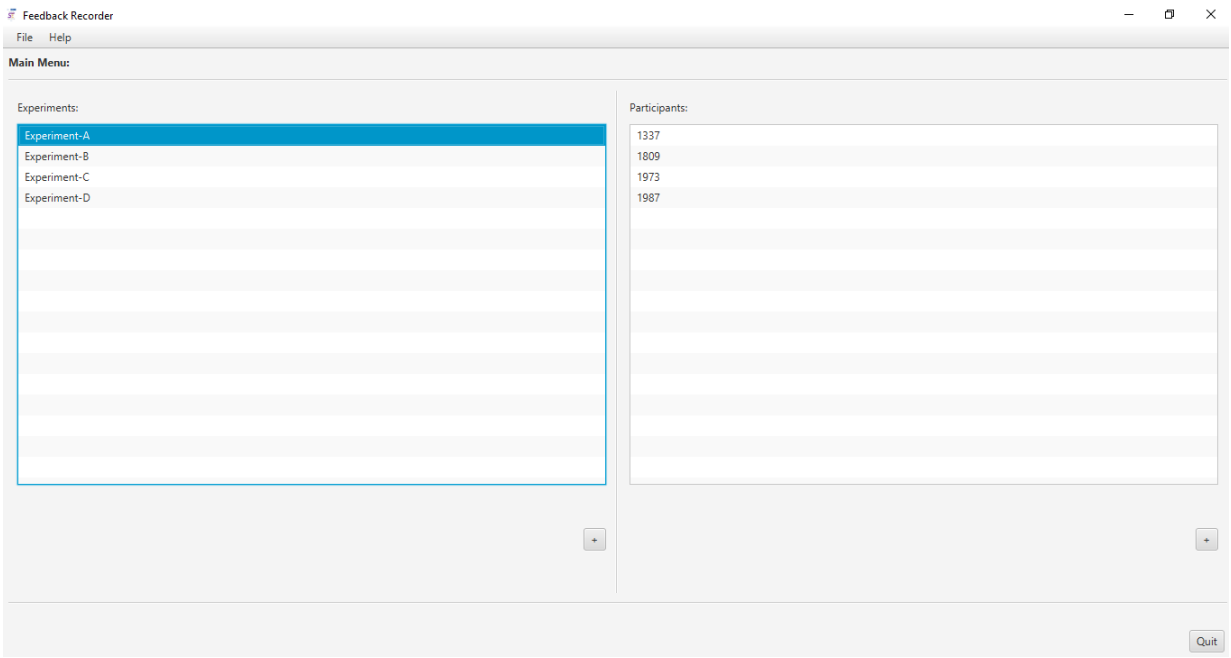


Figure A.1: Screenshot of *Main Window*

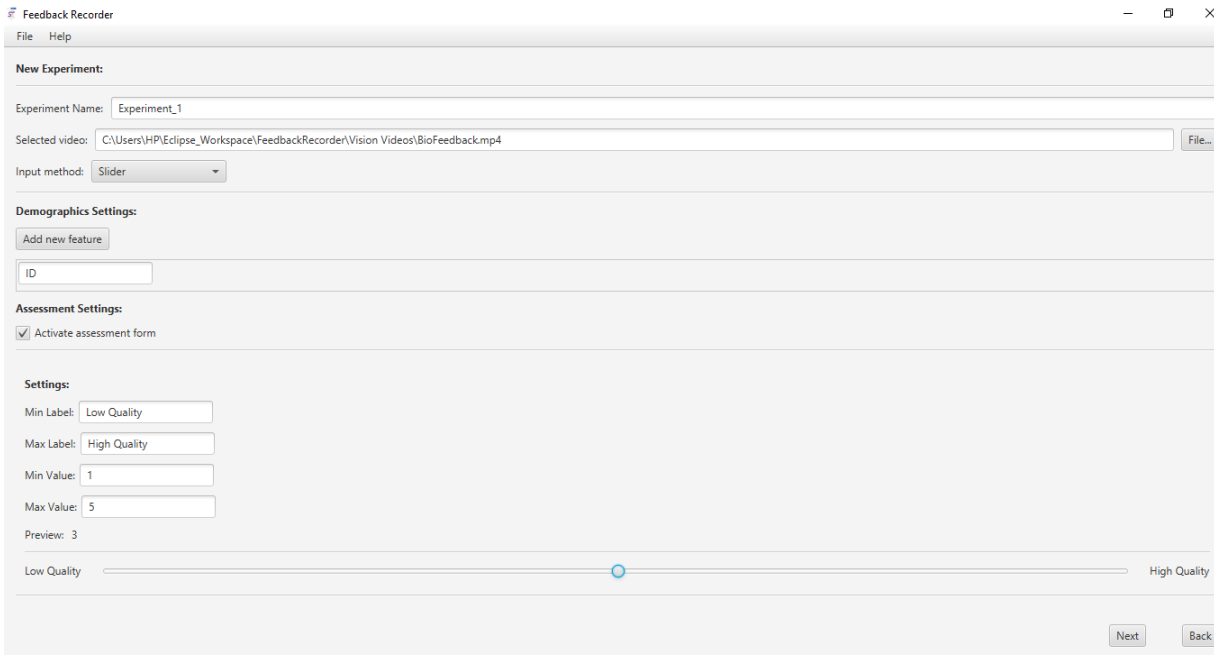


Figure A.2: Screenshot of *Experiment Window* with Slider

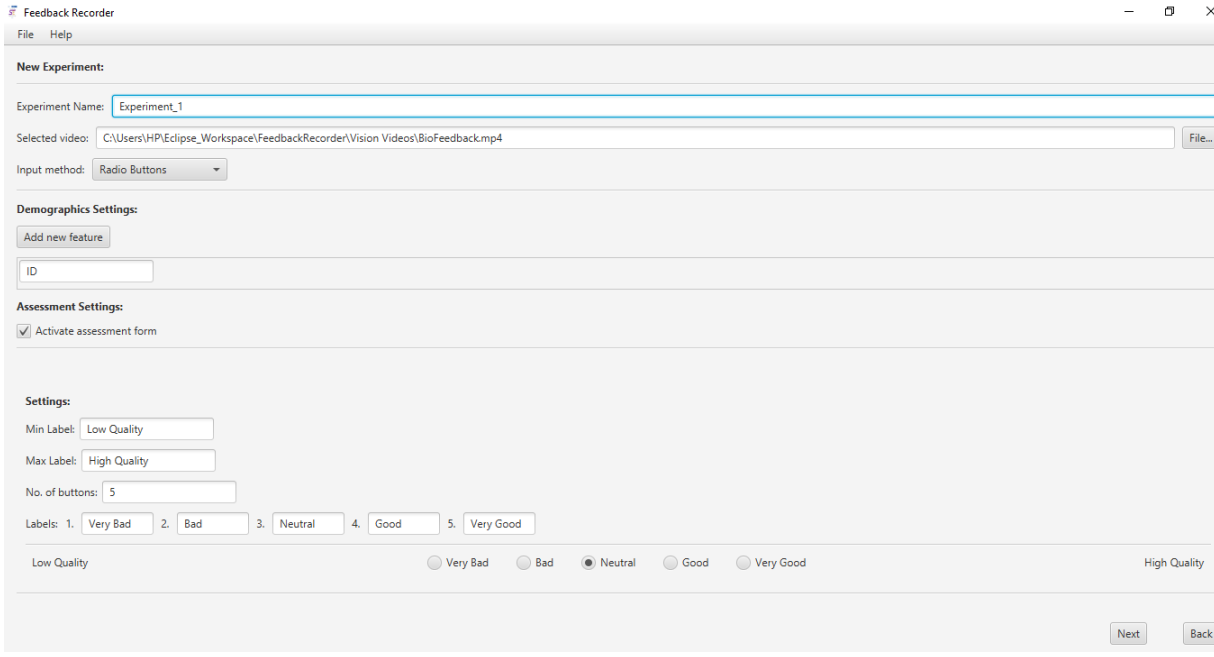


Figure A.3: Screenshot of *Experiment Window* with Radio Buttons

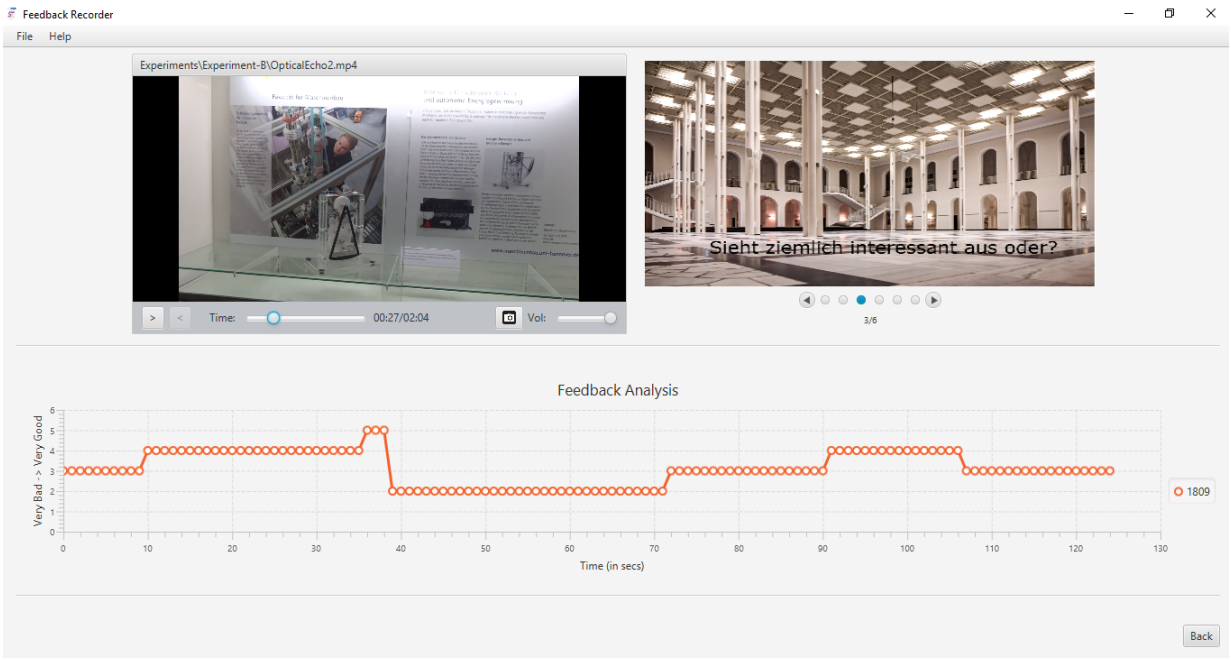


Figure A.4: Screenshot of *Data Graph Window* for a Single Participant

Figure A.5: Screenshot of *Data Graph Window* for Multiple Participants of an Experiment

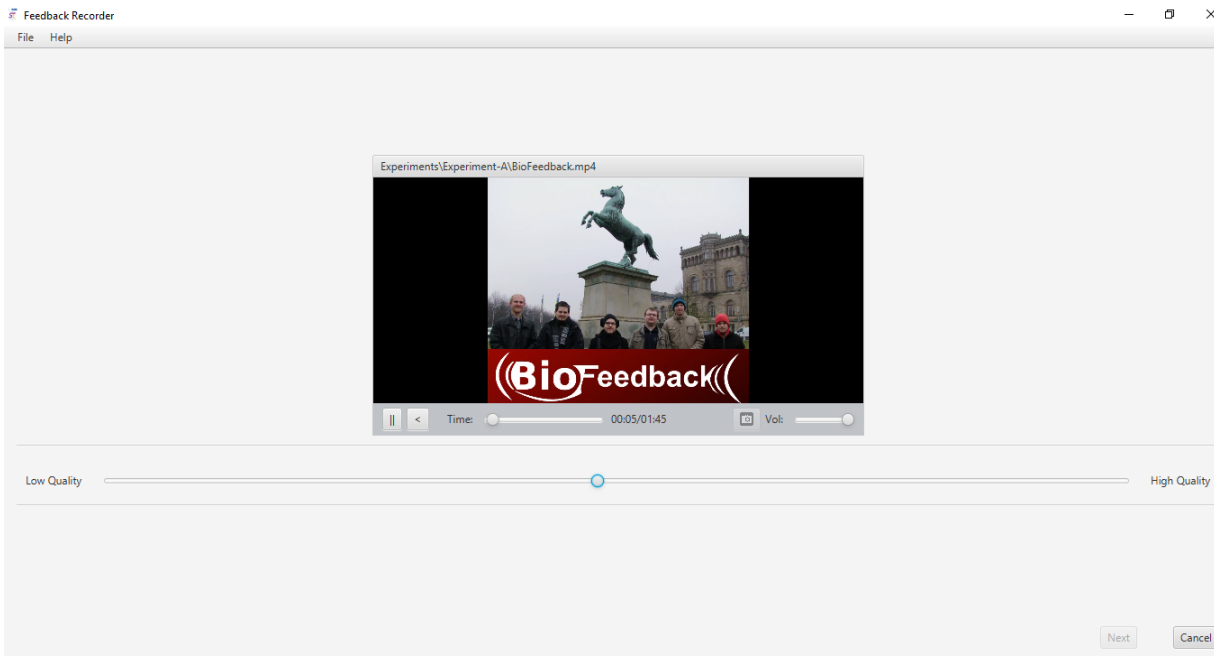


Figure A.6: Screenshot of *Recorder Window* with Slider

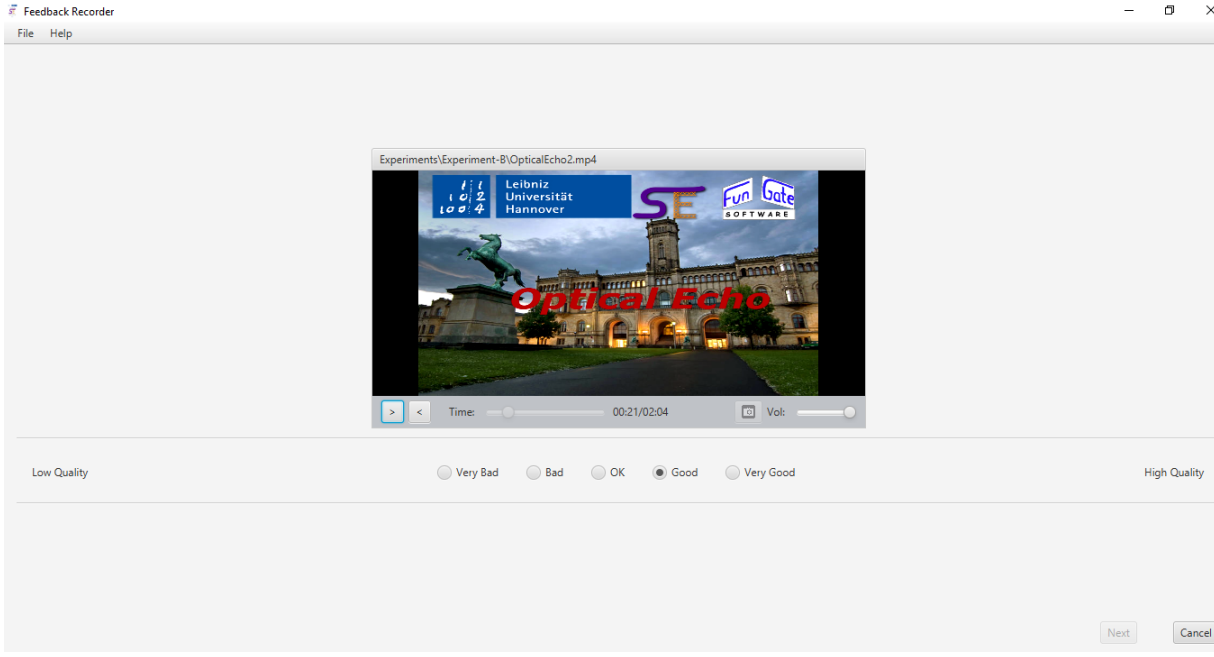


Figure A.7: Screenshot of *Recorder Window* with Radio Buttons

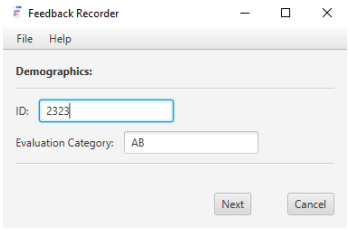


Figure A.8: Screenshot of *User Info Window*

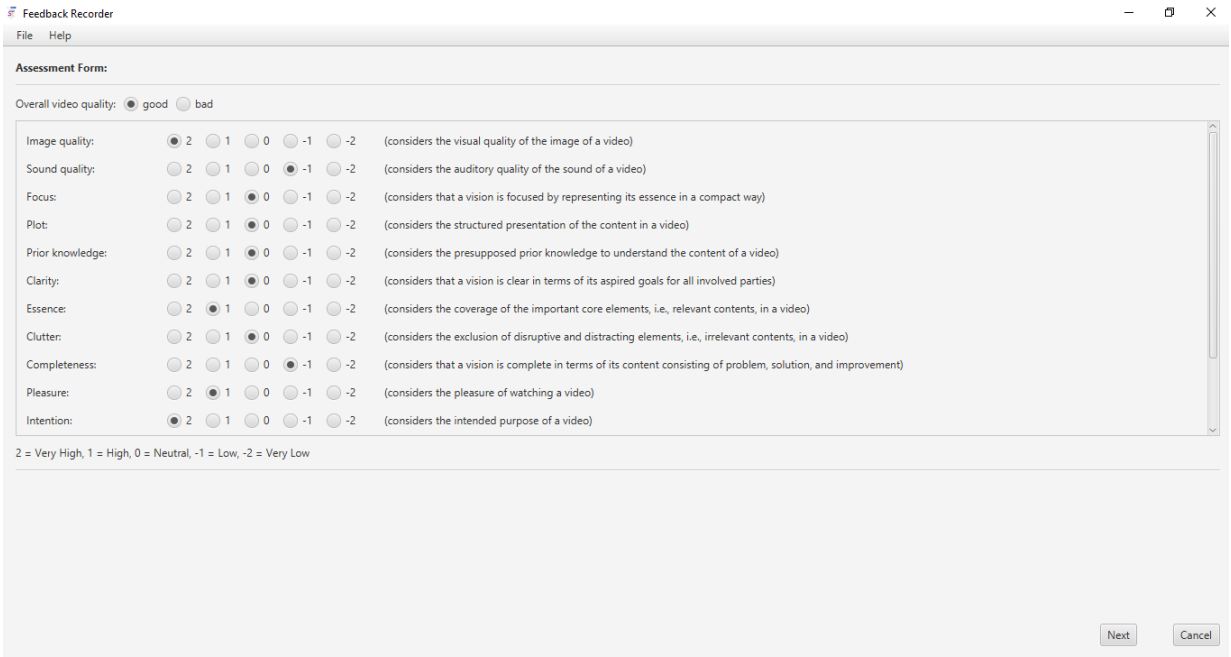


Figure A.9: Screenshot of *Question Form Window*

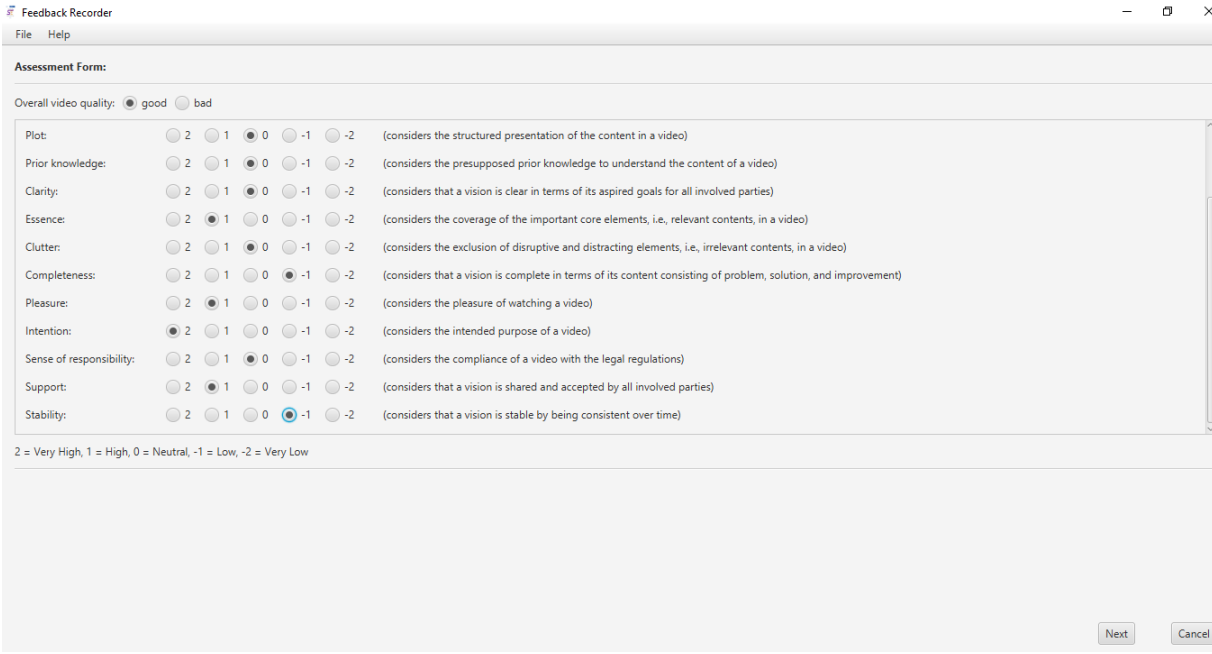


Figure A.10: Screenshot of *Question Form Window* (scrolled down)

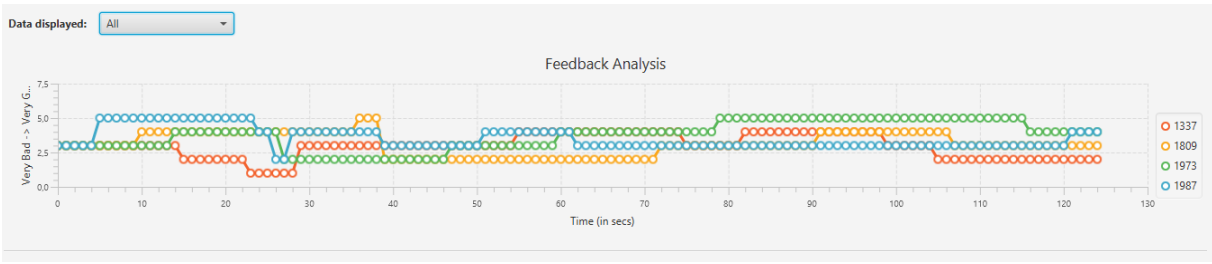


Figure A.11: Screenshot of *Data Analyzer* in **All** Mode

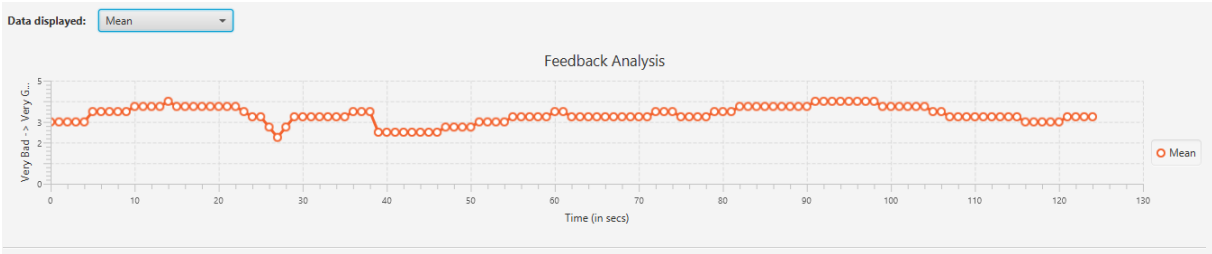


Figure A.12: Screenshot of *Data Analyzer* in **Mean** Mode

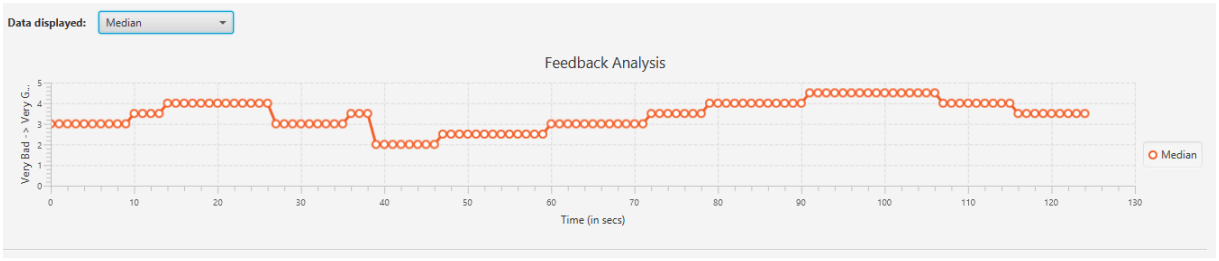


Figure A.13: Screenshot of *Data Analyzer* in **Median** Mode

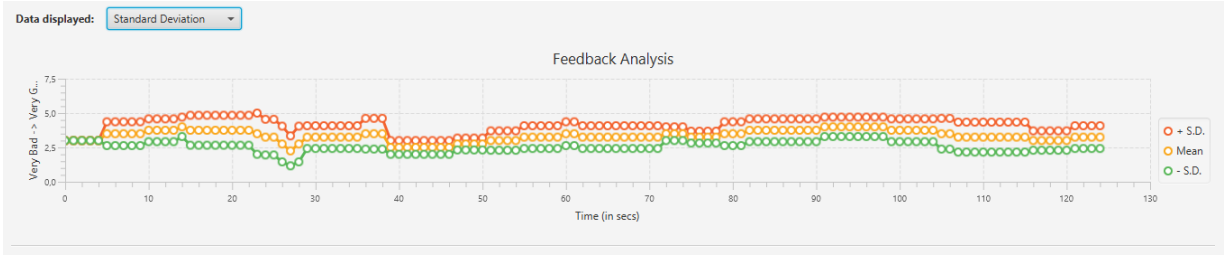


Figure A.14: Screenshot of *Data Analyzer* in **Mean (+/-) Standard Deviation** Mode

A.2 Use-Case-Diagram

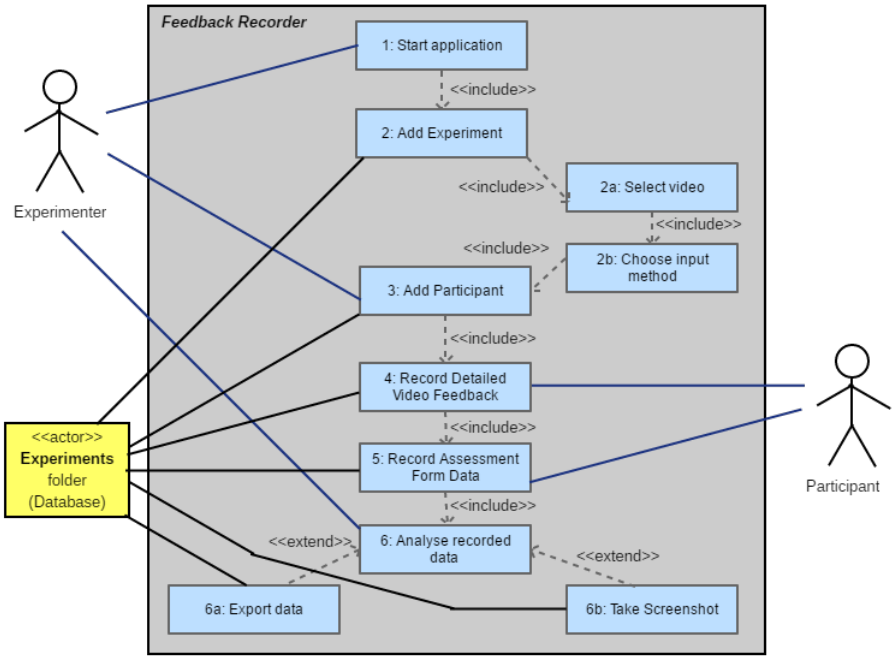


Figure A.15: Use-Case-Diagram of *Feedback Recorder*

A.3 Package structure of application

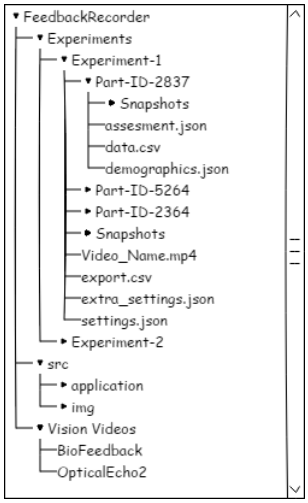
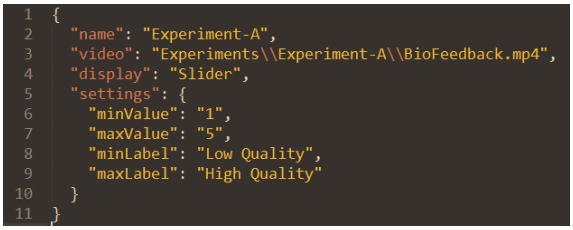


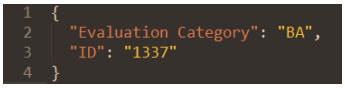
Figure A.16: Package Structure of *Feedback Recorder*

A.4 JSON examples



```
{
  "name": "Experiment-A",
  "video": "Experiments\\Experiment-A\\BioFeedback.mp4",
  "display": "Slider",
  "settings": {
    "minValue": "1",
    "maxValue": "5",
    "minLabel": "Low Quality",
    "maxLabel": "High Quality"
  }
}
```

Figure A.17: JSON Example - *settings.json*



```
{
  "Evaluation Category": "BA",
  "ID": "1337"
}
```

Figure A.18: JSON Example - *demographics.json*

```
{
  "Sense of responsibility": "2",
  "Stability": "0",
  "Support": "0",
  "Overall video quality": "bad",
  "Plot": "1",
  "Prior knowledge": "0",
  "Pleasure": "1",
  "Sound quality": "0",
  "Clarity": "1",
  "Clutter": "1",
  "Image quality": "0",
  "Focus": "0",
  "Completeness": "2",
  "Intention": "1",
  "Essence": "1"
}
```

Figure A.19: JSON Example - *assessment.json*

A.5 Survey Setup

	Slider (1-5)	Radio Buttons (1-5)
BioFeedback	A	C
OpticalEcho2	D	B

Participant	First Video	Second Video
P1	A	B
P2	C	D
P3	B	A
P4	D	C
P5	A	B
P6	C	D
P7	B	A
P8	D	C

Figure A.20: Survey Setup - Mapping of *Evaluation Categories*

Part No.	ID	Eval Cat.	Student?	SWT?	SWQ?	SWP?	Prog. Kenntnisse?
P1	1987	AB	Ja	Ja	Nein	Nein	Ja
P2	101	CD	Ja	Ja	Nein	Nein	Ja
P3	1973	BA	Ja	Ja	Nein	Nein	Ja
P4	5837	DC	Ja	Ja	Ja	Nein	Ja
P5	1809	AB	Ja	Ja	Nein	Ja	Ja
P6	1994	CD	Ja	Ja	Ja	Ja	Ja
P7	1337	BA	Ja	Ja	Ja	Ja	Ja
P8	1020	DC	Ja	Ja	Nein	Ja	Ja

Figure A.21: Survey Setup - *Participant Demographics*

Experiment Overview

Topic Name: Tool-Supported Data Collection for Experiments to Subjectively Assess Vision Videos
Experimenter: Vignesh Arulmani Sankaranarayanan
Institute: Software Engineering Institute, Leibniz Universität Hannover

Please read through the overview carefully. Please feel free to ask questions if any of the given information is unclear.

Firstly thank you very much for participating in our survey. The data collected from the several participants through this particular survey helps us evaluate and subjectively assess the quality of Vision Videos in a detailed manner by taking the temporal changes in the detected values and also other video quality metrics into account.

- Two different Vision Videos would be shown to you, each with a different input method (i.e. 'Slider' or 'Radio Buttons'). While viewing a video you get to record how you feel about the quality of the video at each and every point - for ex., you can drag a Slider more to the right if you find the video to be of a good quality at a particular time and more to the left instead if the video is bad (Similarly you change the values by clicking on a new button, if you are using Radio Buttons instead).
- After which you would get to fill an assessment form electronically, where you can provide your ratings to the video with respect to different video quality aspects.
- Finally a printed Additional Survey form (with demographics and minor questions regarding the experiment) would remain to be filled, marking the end of the experiment.

The survey would take roughly around 20 minutes. You have the right to cancel or withdraw your participation from the experiment at any point during the experiment and with no need for an explanation.

Figure A.22: Survey Setup - *Experiment Overview*

Consent Form

Topic Name: Tool-Supported Data Collection for Experiments to Subjectively Assess Vision Videos
Experimenter: Vignesh Arulmani Sankaranarayanan
Institute: Software Engineering Institute, Leibniz Universität Hannover

1. Physical Condition

"I currently have no physical complaints oder pains."
☐ Yes ☐ No

2. Data Privacy Policy

"I was informed that the data collected during the study would be indefinitely stored and would be used for the evaluation of the survey. The recorded data would be used for scientific research and would be made sure to be evaluated and used in an explizitly anonymous manner. Additionally the data would be published in scientific publications anonymised."

For questionnaires you also have the right to information and deletion of your personal data in accordance with data protection. You can revoke this declaration of consent at any time. After the revocation your personal data will be deleted and will not be used for any further publications.

"With the information mentioned under point 2 (Data Privacy Policy), I"
☐ Agree ☐ Do not Agree

3. Signature

"I have read and understood the overview of the survey. I am participating in this experiment of my own freewill and without any kind of remuneration and feel healthy enough to be able to take part."

Name, First Name (of Participant)

4-digit ID (chosen by Participant)

Place, Date, Signature of Participant

Figure A.23: Survey Setup - *Consent Form*

Additional Survey

1. Demographics:

- ID: _____
- Evaluation Category: _____
- Vocation:

  ☐ Student ☐ Other: _____
- If student, which of the below subjects have you passed:

  ☐ Software-Technik (SWT) ☐ Software-Qualität (SWQ) ☐ Software-Projekt (SWP)
- Do you have a basic experience in programming:

  ☐ Yes ☐ No

2. Session-based Questions:

- Better input method: ☐ Slider ☐ Radio Buttons

  Reason: _____
- Was the recording process distracting?

  ○ Strongly Disagree ○ Disagree ○ Neutral ○ Agree ○ Strongly Agree

  Reason: _____

Figure A.24: Survey Setup - *Additional Survey*

3. Tallies:

- Number of times REWIND used:

  _____ ☐

- Number of times value changed during recording:

  _____ ☐

Figure A.25: Survey Setup - *Additional Survey* (backside)

A.6 Survey Results

Slider/RB?	Recording Process Disturbing?	#(REWIND)	#(VALUE CHANGES)
RB	Strongly Disagree	0	8 (A), 7 (B)
Slider	Disagree	4	8 (C), 7 (D)
Slider	Strongly Disagree	1	6 (B), 2 (A)
Slider	Strongly Disagree	0	6 (D), 8 (C)
Slider	Strongly Disagree	1	2 (A), 5 (B)
Slider	Disagree	0	5 (C), 10 (D)
Slider	Disagree	0	6 (B), 2 (A)
Slider	Neutral	0	2 (D), 1 (C)

Figure A.26: Survey Results - *Additional Survey*

CHARACTERISTIC	1987 (A)	1987 (B)	101 (C)	101 (D)
Overall video quality	good	good	good	good
Image quality	2	1	1	0
Sound quality	1	2	0	1
Focus	-1	0	0	1
Plot	0	1	-2	1
Prior knowledge	-2	-2	-1	1
Clarity	-1	-2	-2	1
Essence	0	0	-1	0
Clutter	1	0	-2	2
Completeness	0	0	-2	0
Pleasure	1	1	-2	1
Intention	2	2	-2	1
Sense of responsibility	2	2	0	0
Support	1	1	0	0
Stability	2	2	-1	2

Figure A.27: Survey Results - *Assessment Data of P1 and P2*

CHARACTERISTIC	1973 (B)	1973 (A)	5837 (D)	5837 (C)
Overall video quality	good	bad	bad	good
Image quality	0	-1	-1	1
Sound quality	1	1	2	2
Focus	0	-1	-1	1
Plot	-1	-2	-1	1
Prior knowledge	0	-1	2	-2
Clarity	1	0	1	1
Essence	1	0	-2	1
Clutter	2	0	-1	-1
Completeness	1	-2	-2	-1
Pleasure	2	-2	1	1
Intention	2	-2	2	2
Sense of responsibility	1	0	0	-1
Support	1	0	-1	2
Stability	2	0	-2	-2

Figure A.28: Survey Results - *Assessment Data of P3 and P4*

CHARACTERISTIC	1809 (A)	1809 (B)	1994 (C)	1994 (D)
Overall video quality	bad	good	good	good
Image quality	0	1	1	1
Sound quality	1	1	0	2
Focus	0	0	0	1
Plot	-1	2	1	1
Prior knowledge	-2	1	1	2
Clarity	-1	0	1	1
Essence	1	1	0	1
Clutter	2	0	0	0
Completeness	-1	1	0	1
Pleasure	0	1	1	1
Intention	0	0	1	1
Sense of responsibility	1	1	0	0
Support	0	0	1	1
Stability	0	2	1	0

Figure A.29: Survey Results - *Assessment Data of P5 and P6*

CHARACTERISTIC	1337 (B)	1337 (A)	1020 (D)	1020 (C)
Overall video quality	bad	bad	good	good
Image quality	0	0	1	0
Sound quality	1	0	-2	-2
Focus	1	0	0	0
Plot	1	1	1	0
Prior knowledge	2	0	0	0
Clarity	-1	1	1	1
Essence	0	1	0	0
Clutter	0	1	0	0
Completeness	0	2	0	0
Pleasure	0	1	0	0
Intention	0	1	0	0
Sense of responsibility	2	2	0	0
Support	1	0	1	0
Stability	-1	0	0	0

Figure A.30: Survey Results - *Assessment Data of P7 and P8*

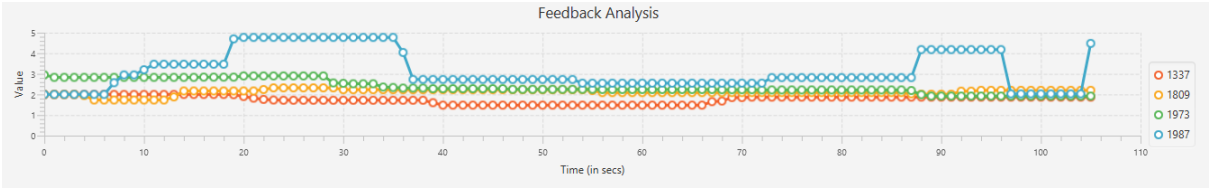


Figure A.31: Survey Results - *Data Graph of* ***A***

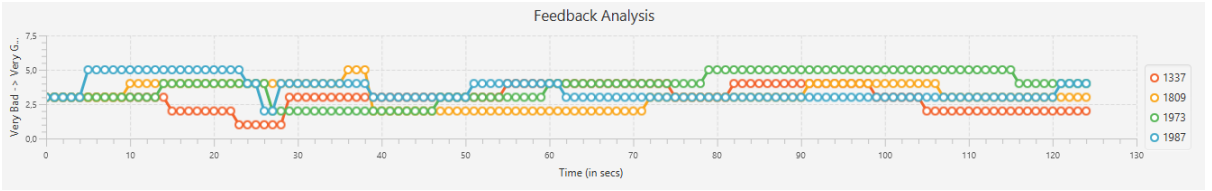


Figure A.32: Survey Results - *Data Graph of* ***B***

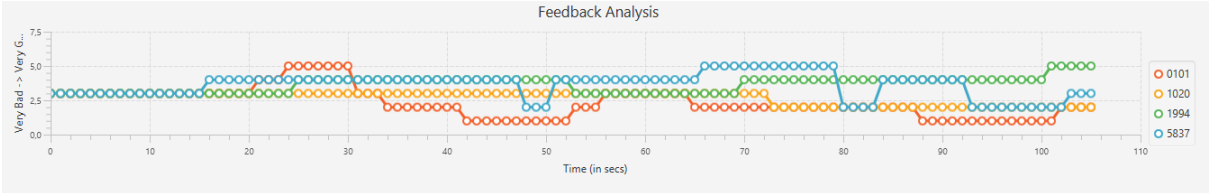


Figure A.33: Survey Results - *Data Graph of* ***C***

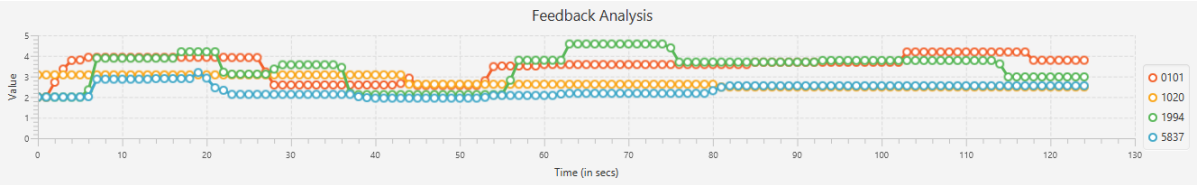


Figure A.34: Survey Results - *Data Graph of* ***D***

List of Figures

Bibliography

[1] S. Ambler. Effective practices for extreme programming and the unified process. *Ist. Ed. John Wiley & Sons, Inc*, 2002.

[2] D. R. Anderson, K. P. Burnham, and W. L. Thompson. Null hypothesis testing: problems, prevalence, and an alternative. *The journal of wildlife management*, pages 912–923, 2000.

[3] V. R. Basili. Software modeling and measurement: the goal/question/-metric paradigm. Technical report, 1992.

[4] B. W. Boehm, J. R. Brown, and M. Lipow. Quantitative evaluation of software quality. In *Proceedings of the 2nd international conference on Software engineering*, pages 592–605. IEEE Computer Society, 1976.

[5] E. Börger, B. Hörger, D. Parnas, and H. Rombach. Requirements capture, documentation, and validation. In *Dagstuhl Seminar*, number 99241. Citeseer, 1999.

[6] V. R. B. G. Caldiera and H. D. Rombach. The goal question metric approach. *Encyclopedia of software engineering*, pages 528–532, 1994.

[7] S. A. Fricker, K. Schneider, F. Fotrousi, and C. Thuemmler. Workshop videos for requirements communication. *Requirements Engineering*, 21(4):521–552, 2016.

[8] K. Holtzblatt and H. R. Beyer. Requirements gathering: the human factor. *Communications of the ACM*, 38(5):31–32, 1995.

[9] D. Jamwal. Analysis of software quality models for organizations. *International Journal of Latest Trends in Computing*, 1(2):19–23, 2010.

[10] S. Jeon, M. Han, E. Lee, and K. Lee. Quality attribute driven agile development. pages 203–210, 08 2011.

[11] O. Karras. Werkzeugunterstützte analyse von requirements-workshop-videos, 2015.

[12] O. Karras, S. Kiesling, and K. Schneider. Supporting requirements elicitation by tool-supported video analysis. In *2016 IEEE 24th International Requirements Engineering Conference (RE)*, pages 146–155. IEEE, 2016.

[13] O. Karras, J. Klünder, and K. Schneider. Enrichment of requirements specifications with videos-enhancing the comprehensibility of textual requirements. 2016.

[14] B. Kitchenham and S. L. Pfleeger. Software quality: the elusive target [special issues section]. *IEEE software*, 13(1):12–21, 1996.

[15] S. Maalem and N. Zarour. Challenge of validation in requirements engineering. *Journal of Innovation in Digital Ecosystems*, 3(1):15–21, 2016.

[16] C. Mergenthaler. Mechanismen zur interaktiven betrachtung von vision videos im requirements engineering. Master's thesis, Leibniz Universität Hannover, Fachgebiet Software Engineering, 2019.

[17] B. Nuseibeh and S. Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 35–46. ACM, 2000.

[18] F. Paetsch, A. Eberlein, and F. Maurer. Requirements engineering and agile software development. In *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.*, pages 308–313. IEEE, 2003.

[19] R. Pham, S. Meyer, I. Kitzmann, and K. Schneider. Interactive multimedia storyboard for facilitating stakeholder interaction: supporting continuous improvement in it-ecosystems. In *2012 Eighth International Conference on the Quality of Information and Communications Technology*, pages 120–123. IEEE, 2012.

[20] S. Radack. The system development life cycle (sdlc). Technical report, National Institute of Standards and Technology, 2009.

[21] M. Schrapel and M. Rohs. Refining vision videos. In *Requirements Engineering: Foundation for Software Quality: 25th International Working Conference, REFSQ 2019, Essen, Germany, March 18-21, 2019, Proceedings*, volume 11412, page 135. Springer, 2019.

[22] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack. Study of subjective and objective quality assessment of video. *IEEE transactions on Image Processing*, 19(6):1427–1441, 2010.

[23] B. Zeiss, D. Vega, I. Schieferdecker, H. Neukirchen, and J. Grabowski. Applying the iso 9126 quality model to test specifications–exemplified for ttcn-3 test specifications. *Software Engineering 2007–Fachtagung des GI-Fachbereichs Softwaretechnik*, 2007.