Graphical Narrative Interfaces: Representing Spatiotemporal Information for a Human-Robot Team with a Highly Autonomous Robot

Hiroaki Nakano
Brigham Young University

Graphical Narrative Interfaces: Representing Spatiotemporal Information for a Human-Robot Team with a Highly Autonomous Robot

Hiroaki Nakano

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Brigham Young University

May 2016

ABSTRACT

Graphical Narrative Interfaces: Representing Spatiotemporal Information for a Human-Robot Team with a Highly Autonomous Robot

Hiroaki Nakano
Department of Computer Science, BYU
Master of Science

Having a well-developed Graphical User Interface (GUI) is often necessary for a human-robot team, especially when the human and the robot are not in close proximity to each other or when the human does not interact with the robot in real time. Most current GUIs process and display information in real time, but the time to interact with these systems does not scale well when the complexity of the displayed information increases or when information must be fused to support decision-making. We propose a new interface concept, a Graphical Narrative Interface (GNI), which presents story-based summaries driven by accumulated data. This thesis (a) uses literature and preliminary GNI designs to identify a set of design requirements for easily managing spatiotemporal information, (b) presents a set of algorithms designed to satisfy these requirements, (c) evaluates the utility and limitations of these algorithms, (d) describes a prototype GNI that combines these algorithms with a graphical interface, and (e) compares the GNI and a GUI through a user study and evaluates the efficiency of the GNI.

Keywords: narrative, interface design, GUI

ACKNOWLEDGMENTS

I would like to express my special appreciation and thanks to my advisor, Professor Michael Goodrich. You have been a tremendous mentor for me and your advice on my research has been priceless. I would also like to thank my committee members, Professor David Wingate and Professor Mark Clement, for serving as my committee members.

Thanks to NASA and ONR for the funding. I also thank Debra Schreckenghost, Todd Milum, and other associates at TRACLabs. At the end I would like to express appreciation to my wife, Takako, and my family members for their support.

Table of Contents

List of Figures **vii**

List of Tables **x**

1 Introduction **1**

1.1 Background 1

1.2 Solution Motivation 3

1.3 Related Literature 4

1.4 Solution Approach 5

1.4.1 Concept of Narratives 6

1.4.2 Narratives and Situation Awareness 7

1.4.3 Application and Design of an Interface 8

1.5 Thesis Statement 10

1.6 Contribution of this thesis 10

2 Implementation Details **12**

2.1 Three Implementation Concepts 14

2.1.1 Anytime Summarization 14

2.1.2 Storytelling 15

2.1.3 Multi-Perspective Analysis 17

3 Model **18**

3.1 Data Block 18

3.2 Block Database 22

3.3 Representing Mission Flow 22

3.4 Comparing *Planned* to *Executed* 25

3.5 Aggregating Information 27

3.5.1 Grading Execution Between An Event And Its Successor 27

3.5.2 Aggregating Across an Execution Trace 28

3.5.3 Discussion of the Aggregation Algorithm 29

3.6 Generating Summaries 29

3.6.1 Higher Levels of Abstraction 31

3.6.2 Detecting Data Mismatch 32

4 View **35**

4.1 Information Presentation 35

4.1.1 Text 36

4.1.2 Images 39

4.1.3 Time Series 40

4.1.4 Geospatial Information 43

5 Controller **45**

5.1 Event Selector 45

5.2 Relations 46

6 Demonstration of the Features of GNI **50**

6.1 Implementation Requirements 50

6.1.1 Requirement 1 51

6.1.2 Requirement 2 51

6.1.3 Requirement 3 54

6.1.4 Requirement 4 56

6.1.5 Requirement 5 56

6.2 Conceptual Interaction 57

7 Validation **61**

7.1 User Study Setup 64

7.2 Procedures 65

7.3 Results and Analysis 66

7.3.1 Demographics 66

7.3.2 Time 67

7.3.3 Situation Awareness 68

7.3.4 Mouse Clicks 71

7.3.5 NASA-TLX 71

7.3.6 Direct Comparison 72

7.4 Discussion 73

8 Summary and Conclusions **74**

9 Future Work **75**

Appendices **79**

Appendix A User Study Questions **80**

Appendix B Information and Questions Given to Participants During Tutorial Session **81**

Appendix C Practice Session Questions **87**

Appendix D NASA-TLX **88**

Appendix E Direct Comparison **89**

References **90**

List of Figures

2.1 Basic layout of components on the GNI. 13

3.1 Structure of a block 19
3.2 First two tasks in JSON format 21
3.3 First Planned Data Block 21
3.4 Second Executed Data Block 21
3.5 Example of the data table 22
3.6 A transition system of a simple beverage vending machine. 23
3.7 A mission plan represented in Kripke Structure. 24
3.8 Possible execution data (single constraint or atomic proposition only) represented in Kripke Structure. 24
3.9 Summary of analyzing and representing data 32
3.10 Overview of the principles of narrative generation and abstraction 33
3.11 Data Mismatch on Plan Kripke Structure 34
3.12 Data Mismatch on Possible Execution Kripke Structure 34

4.1 Story Generation Button 36
4.2 Enlarged Text Box 37
4.3 Example of the Image Display Console 40
4.4 Explanation of the icons used on the Time Indicator 40
4.5 Time Indicator Magnification Example 41
4.6 Linked icons on Time Indicators. 41
4.7 Screen Shot of a Map of the GNI. 43

4.8 List of the Icons Used on the Map. 43

5.1 Enlarged Event Selector 45
5.2 Use of Event Selector 47
5.3 Before and After of Mouse Click 47
5.4 Event Details View 48

6.1 Example of summary at level 1 52
6.2 Example of summary at level 2 52
6.3 Example of summary at level 3 53
6.4 Example of summary at level 4 53
6.5 Example of summary at level 5 54
6.6 Example execution of a short mission 55
6.7 Example execution of a mid-length mission 55
6.8 Example execution of a long mission 56
6.9 First Interaction 58
6.10 Generated Summaries at Different Levels 59
6.11 Selection of Tasks To Be Displayed 60
6.12 Finding Icons For A Task 60

7.1 Two Interfaces Used For User Study 62
7.2 List of Icons Used for User Study 62
7.3 Averaged Time For Each Interface To Complete The Entire Session In Minutes 67
7.4 Averaged Time for Each Interface for Native Language 68
7.5 Averaged Time for Each Interface for Gender 68
7.6 Averaged Number of Questions Answered Correctly for Interfaces 69
7.7 Averaged Number of Questions Answered Correctly for Gender 69
7.8 Averaged Percentage for each Question for Each Interface 70
7.9 Averaged Time to Answer Questions for Different Interfaces 70

7.10 Averaged Mouse Click for Question 5 for Native Language 71
7.11 Averaged Mouse Click for a Session for Gender 71
7.12 NASA-TLX 72
7.13 Direct Comparison 72

9.1 See-ability as an indexing cue. The visual indicator bar indicates which video segment includes the largest point (green marker) and the useful resolution of those segments Shading indicates quality, from bright yellow (high quality) to black (not seen). Clicking on this indicator moves directly to those frames. . 76
9.2 An example of a perspective display. The Unmanned-Aerial-Vehicle (UAV) is displayed in the center and heading direction and other information is also displayed accordingly. 77
9.3 Coverage quality map overlaid in red on terrain and reference imagery. The UAV was launched from the uphill side of the terrain (orange maker on the left) and followed the flight path indicated in green. 77

List of Tables

3.1 A table to combine outputs for two constraints into one. 31

6.1 Runtime Comparison Results. Number unit is seconds. 57

7.1 Starting Conditions 64

7.2 Statistics of the Participants of the User Study 66

7.3 Distributions of Users With Different Conditions For Each Group 66

Chapter 1

Introduction

1.1 Background

Human-robot teams flourish in many different fields and are used for diverse purposes [12, 18, 43], but the methods of communication between humans and robots differ across applications [53]. Humans and robots may communicate through gestures, voice, sounds, or computer graphics. Since the best combination of these elements depends on the specific application, it is useful to introduce the application scenario that is the focus of this thesis:

An automated rover is placed on another planet. Astronauts are supervising the rover from the planet's orbit. Information transmitted from the rover will arrive at the ground base on earth after passing through the orbiter occupied by the astronauts. The rover is used to surveil potential antenna sites and to prepare the ground for an eventual extra-vehicle geology expedition by astronauts. While the rover is executing a mission plan, various problems may occur: malfunctioning equipment, obstacles in the rover's path, communication loss, and so forth. Human users, both astronauts and members of the ground crew on earth, need to be aware of these situations and make adjustments to the plan so that the mission will be accomplished.

In the space-based application area described above, the robot will operate in a remote area so having a GUI is a natural way (a) to represent spatiotemporal information obtained by the vehicle and (b) to enable the user to interact with this information to make sense of the remote environment.

Naturally, the best form of interaction depends not only on the application but also the capabilities of the robot. Teleoperation occurs when the robot has very low autonomy, but this form of interaction is inappropriate for a space-based application because of the potentially high time delays in communication. Supervisory control, by contrast, occurs when the robot is given sufficient autonomy to perform a suite of tasks under human supervision and is more appropriate for applications where time delays can occur [47].

Importantly, if the robot is capable of operating for periods of time without direct human input, the human has what Olsen has called *spare capacity* in which it is possible for a human to oversee multiple tasks [11, 37]. One way for a human to use this spare capacity is to supervise multiple robots, which was how Olsen used it. Another way for the human to use this spare capacity, which is the focus of this thesis, is for the human to manage a single robot but use that robot to perform a complicated task that requires many different subtasks to be performed, some by the human and some by the robot; NASA's Mojave Volatiles project is a good example [46]. This thesis focuses on a single human user and a single robot for a mission that consists of multiple sequential tasks.

Although the robot can be "neglected" during periods of autonomous operation, robot performance and behavior eventually deviates from what is desired; how far performance drops depends on the level of autonomy and the conditions of the environment. *Neglect time* has been defined as the expected amount of time that a robot can be ignored before its performance drops below a threshold [11]. *Interaction time*, the complement to neglect time, has been defined as the expected amount of time that a human must interact with a robot to bring it to an acceptable level of performance. Prior work has indicated that supervisory control in human-robot teaming benefits when the system is designed to increase neglect time and to decrease interaction time [11].

This thesis assumes that the robot's autonomy is such that neglect time is fixed and, consequently, the thesis discusses ways in which interaction time can be minimized. Decreasing interaction time can be beneficial for many problems, but most current GUIs do not directly address the interaction time problem because they are focused on real-time processing and control. This means that those interfaces are designed to display information when they obtain it, assuming that users will see the information immediately. This is a poor assumption for a human-robot team designed for the scenario described previously because, by design, the ability to neglect a robot while performing other tasks means that attention must be shared between multiple tasks, making it likely that changes in information may not be perceived or understood by the human manager [20, 49]. Importantly, emphasizing interaction time opens up the possibility of applying the GNI concept to post-mission evaluation and problems associated with human-data interactions.

1.2 Solution Motivation

It should be possible to design interfaces that support attention management and decision-making for complicated tasks like the one described in the scenario above. Although GUIs are designed to facilitate easy acquisition of situation awareness [16], they often fail to explicitly represent the mission-based narrative that is necessary for understanding how current data relates to previous observations and future objectives. This is especially true when there is a time gap between the display of information on the interface and the observation of the displayed information by the user. Metaphorically, the information displayed by a traditional GUI acts like a list; the GUI displays different kinds of useful information but fails to "gather up and make sense" of the information as a whole. Alarms [21], decision support systems [24], ecological displays [36], maps [32], change summaries [51], and other technologies can be used to make GUIs more effective, but we propose a more holistic approach.

For example, if an installed video camera on an automated rover stops working, a typical GUI might (a) display an alert or an alarm to explicitly let users know about the problem or simply (b) display a blank screen on the interface. The incident may be recorded in a log as well. Suppose that the camera failed for only a short period of time and quickly resumed working properly. It is possible and maybe even likely that the human missed the alarm because the user's attention was away from the interface when the incident happened and may be completely unaware of the incident until the log is reviewed as part of the mission debriefing. During the debrief, the user will find a record of the incident that the camera went off for a while and needs to figure out both why the incident occurred and what possible effects the incident had on the mission (e.g., was critical information missed).

The key observation from this example is that, although the GUI may provide cues intended to guide the human's attention in real time, it is ultimately the user's responsibility to recognize the problem, put it in context, and appropriately respond to it. The consequence of this is that users are forced to closely monitor robots during a mission to grasp the entire narrative "as executed" and handle all problems for which "as executed" behavior fails to match "as planned". Consequently, interaction time can be high because the human must mentally represent the mission-based narrative in order to gain situation awareness.

This thesis presents a prototype GNI that consists of a set of algorithms that allow for the representation and summary of a narrative constituted by comparing a planned mission to an executed mission.

1.3 Related Literature

Interface design and development for human-robot teams are in transition from targeting low-autonomy robots to targeting high-autonomy robots. Increasing a robot's autonomy causes a change in the human role in a human-robot team [31, 38]. Some interfaces designed for supervisory control tasks are connected to a data storage system that can retrieve data for later use [56]. Interestingly, simple playback of the old information may not help much [35]. Similarly, presenting data faster than real-time [26] to support human supervisors [4, 25] may overwhelm users as they try to extract needed information.

A missing element of playback is that key information, like when and how the accident happened, requires the human to review the entire stored record. "After action reviews" or "change summaries' could potentially help [14, 50] but are often used only after the execution of a mission is complete rather than during mission operation. One reason for this is the assumption that communication between humans and robots is one-way, meaning that robots throw a bunch of perceptual data at humans and leave it up to the users to interpret and understand what those data mean in the mission context. It is better (a) to have two-way interaction between data from robot and humans (similar to the way humans use two-way communication when they interact with each other) and (b) to present, explicitly, summary information that provides trends in the narrative. The conceptual GNI proposed in this thesis attempts to do this.

There is a good example of how narratives are used as useful means of communication in real situations. Fiore's work demonstrates practical uses of narrative on interfaces among human users. In [17], narrative is used to help scattered groups of humans to communicate more efficiently. In [23], narrative is used to enhance the efficiency of debriefing between individuals and groups after a mission by allowing them to share interpersonal knowledge.

The basic framework of our interface is built upon the framework of an existing interface called RESCHU developed by a team at MIT[1]. RESCHU was designed to measure a human user's workload to manipulate multiple robots to complete some tasks. RESCHU is equipped with a map, a timeline, a sensor display, and a panel to view the status of the robots. Our interface has some of these interfaces but not all. We will explain more about the actual implementation of our interface in Chapter 2.

1.4 Solution Approach

We propose a new type of interface that we call a Graphical Narrative Interface (GNI). We claim that a GNI will enable users of highly autonomous robots to complete their missions more efficiently than traditional GUIs. The GNI approach is designed to address problems found in prior GUI designs by enabling a new approach to abstracting data and interactively summarizing narratives. The key GNI concept is, as the name indicates, narratives. Narratives have the capability to express dense information in a way that is easier for readers to understand than other methods. To understand the potential for including storytelling in the GNI, it is useful to understand the effectiveness of storytelling as a communication method [30, 48, 58]. We compare our GNI against another GUI to see if the GNI is actually more useful than the other through a user study. We discuss this in Chapter 7.

1.4.1 Concept of Narratives

Worth says that a "narrative is not merely a list or series of events or states of affairs, but requires some sort of sequence of events, where the sequence minimally implies a temporal ordering" [58]. This "temporal ordering" also includes the manifestation of the connections of the events [59]. This means a narrative is not just a sequenced list of events like "A, B, C", but rather "A happened first, then caused B to happen. D was expected but C ended up happening because ..." Notice that the temporal ordering can include causality.

Characters and their roles also have important meanings [52]. The reader may better perceive the meaning of a narrative if it is told from different perspectives. For example, a story of the race of a hare and a tortoise can be told as follows: "The tortoise won the race because it persistently tried its best regardless of its inferiority in the physical ability". However, the same story can be viewed from another perspective: "The hare lost the race because it slept too long in the middle of the race." In a Human-Robot-Team (HRT), there are at least two characters: a human user and a robot. If there are people who work with the robot in the field, then they are also characters who appear in the narrative. Additionally, there may be multiple roles for human users: mission planner, robot controller, mission supervisor, data analysis, etc. [19, 43].

Another critical element of narratives is the degree of abstraction. When someone is asked about the content of a movie, the person would not take two hours to tell the entire story from beginning to end. Rather, the story is summarized. The length and the content of the summaries varies depending on the purpose [28]. Storytelling is a useful metaphor for understanding how to summarize a narrative since it is easy for storytellers to organize and share information like when, where, and what happened. It also helps listeners to understand what is being told.

The principles of narratives discussed above are 1) temporal ordering, 2) characters and roles, and 3) abstraction. Are these principles of narratives useful and helpful in a real situation where a HRT would be used? In other words, how can we effectively generate narratives out of the work of a HRT?

GNI design principles can be distilled from the literature on storytelling, specifically by techniques introduced in [54, 55]. This literature suggests three essential elements for creating an effective narrative of a HRT mission: 1) *selecting data boundaries*, 2) *incorporating grammatical semantics*, and 3) *presenting results in a context appropriate for the story.* Selecting data boundaries means separating the mission into smaller segments and treating them as individual events. In our project, we treat each task that is part of the robot's plan as a single event. We explain this in more details in the following chapter. Introducing grammatical semantics means generating text that includes the information from events with a temporal order and connections. Presenting results in a context appropriate for the story means generating and displaying narratives with various degrees of abstraction from multiple perspectives.

1.4.2 Narratives and Situation Awareness

There are many similarities between narrative-based understanding and Endsley's conceptualization of Situation Awareness (SA). Endsley proposed that SA, a human's sense of what is going on in the world, consists of three elements [15]:

> ... the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future.

Hone *et al.* simplified Endsley's definition of SA into three general questions: who is where, what are they doing, and what will they do [22]. These are the questions that the human users of automated robots would like to answer to evaluate the level of success of their missions.

Narratives give answers to these questions. Each narrative clearly states who the characters are because that is a basic requirement of a narrative [59]. When the abstraction level is high, a narrative should focus on a few key mission events, and this gives the user a brief understanding of the entire mission. When the abstraction level is low, a narrative should give detailed information about many different parts of the mission. Through temporal ordering, the user can a) learn the relationships and connections between events, b) derive the reason why things went a certain way and c) predict what may happen next.

1.4.3 Application and Design of an Interface

Given this understanding, it is necessary to determine how to apply the concept of a narrative to an actual interface design. This thesis present details, operational definitions, and implementations of components of a prototype GNI in Chapter 2. Before doing so, we list two assumptions about the project domain and mission.

The first assumption is that the GNI will be designed to support a ground robot, which we will call an Unmanned Ground Vehicle (UGV) to emphasize that this is a vehicle-type robot rather than a humanoid or aerial robot. As discussed in Chapter 7, we have access to data gathered by a physical UGV used in a real field operation. The mission of this field operation is for the UGV to traverse through some locations and performs various tasks in a specified order. This problem domain suggests that the GNI will require some kind of map that displays the geographical information as well as some kind of a time indicator to refer to the timing and sequence of events are necessary [10, 29, 57].

The second assumption is that the GNI will use only graphical information and interaction through keyboard and mouse [42, 47]. This means that we do not include sounds, haptic signals, or any other means of communication on our interface. We adopt the standard model-view-controller architecture for a graphical display [27] and include the ability to abstract and display abstractions derived from the model in the GNI.

In contrast to existing GUIs, the GNI focuses on automated data analysis and abstraction of data. This means that users mainly analyze and understand the data collected by the robot and make plans for the future, instead of controlling the actions of the robots throughout a mission. Naturally, the GNI should be able to process and analyze narrative summaries in real-time, and present these summaries in a format that allows the human to make inferences and decisions faster and better than they could do with a traditional GUI.

We distill the narrative concepts and our assumptions about environment and mission into five implementation requirements:

1. The GNI should have a chat-like text console to display auto-generated narrative summaries.
2. The GNI should have the capability to generate narratives at various granularities.
3. The GNI should have a map and a time indicator to handle spatiotemporal information.
4. The GNI should maintain cohesion between components described in Requirements 1 and 2.
5. The GNI should process and display data in real-time.

Requirement 1 is necessary since it is the main concept of GNI. We add a new feature of generating narratives to an existing GUI, which handles spatiotemporal information. However, simply adding an ability to generate narrative may not work efficiently. Requirement 2 adds practicality and usability to the narratives generated by the GNI by adjusting the amount of information according to the user's needs. Requirement 3 is essential because of the type of missions and the nature of data associated with the missions we chose to use. The main purpose of the missions is to explore certain areas and to perform some activities in temporal order. Thus, a component like a map to display spatial information and a component like a time-line to display temporal information are required. Requirement 4 requires a scheme to connect all the components on the GNI so that they work together to tell stories of the missions. However, if a narrative generation process takes too much time, it can be a distraction to a user instead of a support. Requirement 5 asks to minimize the time to generate and display narratives on the GNI so that it will not bother the user to study the data about a mission.

Note that the GNI is not a direct extension of a certain kind of research or study, but rather the GNI is based on a collection and collaboration of several different ideas. This means that the idea of using a timeline, map, and chat window need not be new or novel, but using them to display narrative-based information in a coherent view is relatively new [17, 23]. Thus, the requirements to use a timeline, map, and chat window are obtained not from first principles but rather because these are the commonly used GUI elements for human supervisory control of a UGV such as those found in Goodrich [19].

1.5 Thesis Statement

A Graphical Narrative Interface (GNI) gives users an ability to understand and analyze data accumulated over time more effectively and efficiently than a traditional Graphical User Interface (GUI). The GNI uses a timeline, a map, and a chat window to display a narrative summary that is generated by a set of algorithms that compare a planned sequence of mission steps to an actual sequence of executed steps.

1.6 Contribution of this thesis

This thesis makes the following contributions:

1. Uses a narrative to organize information exchanged in a Human-Robot team.
2. Implements a narrative-based interface by:
    - Formulating a set of algorithms to compare and analyze input data.
    - Finding a way to present narratives using text and visual aids.

- Evaluating effectiveness of the implemented interfaces through a user study.

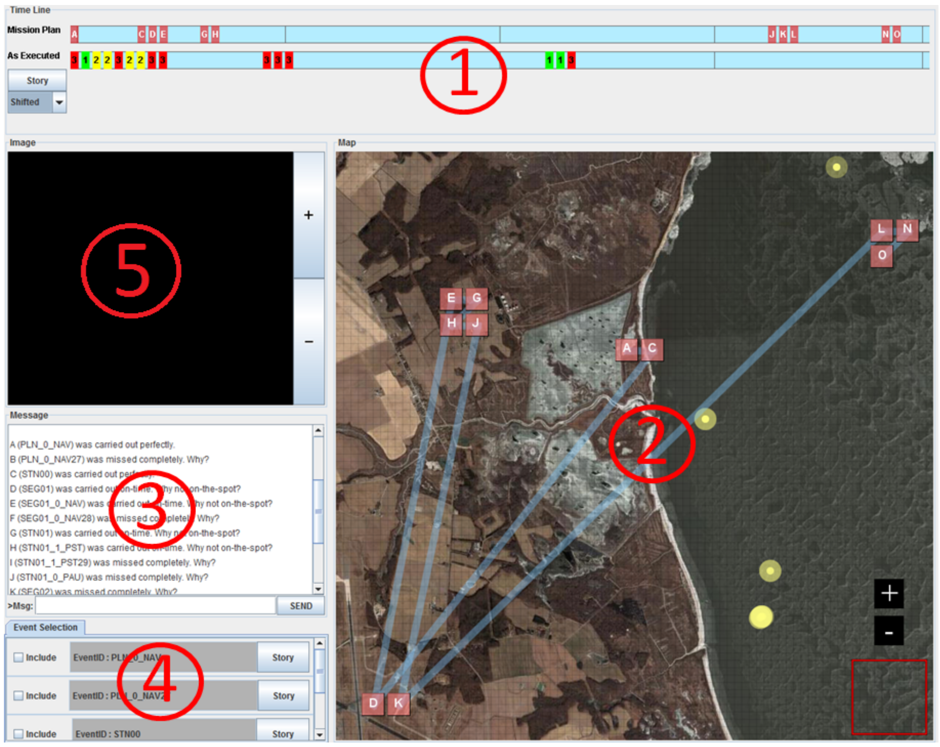
Chapter 2

Implementation Details

In the previous chapter, we described a target human-robot interaction environment and how the concept of narratives can assist a human user to understand the environment. We now move onto a discussion about the details of how we apply the concept of narratives into the implementation of an interface. In this chapter, we briefly explain the connection between this and the next three chapters, and then we present three key implementation concepts of the GNI. The following three chapters present the implementation details, the technical contributions of a Graphical Narrative Interface (GNI), and the connection to the concept of narratives based on the model-view-controller scheme. Note that this GNI is a proof of concept, meaning that future designs are likely to modify, delete, and add needed features.

The structure of this proof-of-concept implementation of the GNI is based on three concepts: *anytime summarization*, *storytelling*, and *multi-perspective analysis*. We explain each concept in the next section. The basic layout of the GNI is shown as Figure 2.1: it has a time indicator labeled as 1, a map labeled as 2, a text console labeled as 3, an event selector labeled as 4, and a sensor display labeled as 5. Details for components 1 through 4 will be given in the following sections. Component 5 displays images and sensor information of the robot. This is a fairly standard element for any type of GUIs and there is no special use on the GNI.

Since it can be argued that the GNI is a specialized type of the more general class of Graphical User Interfaces (GUI), it is useful to relate the standard model-view-controller [27]



```
A (PLN_0_NAV) was carried out perfectly.
B (PLN_0_NAV27) was missed completely. Why?
C (STN00) was carried out perfectly.
D (SEG01) was carried out on-time. Why not on-the-spot?
E (SEG01_0_NAV) was carried out on-time. Why not on-the-spot?
F (SEG01_0_NAV28) was missed completely. Why?
G (STN01) was carried out on-time. Why not on-the-spot?
H (STN01_1_PST) was carried out on-time. Why not on-the-spot?
I (STN01_1_PST29) was missed completely. Why?
J (STN01_0_PAU) was missed completely. Why?
K (SEG02) was missed completely. Why?
```

```
EventID : PLN_0_NAV
EventID : PLN_0_NAV27
EventID : STN00
```

Figure 2.1: Basic layout of components on the GNI.

architecture from GUIs to the GNI design. A model-view-controller architecture aligns perfectly with the GNI design and it can be applied to explain how the five requirements, which we presented in the previous chapter and re-listed below, can be satisfied in the implementation. In the model-view-controller architecture, the "model" includes the data structure and algorithms used to organize and interpret information collected from a robot; we describe the model in Chapter 3. The "view" includes how information is presented to a user, and includes the three key elements mentioned in the thesis statement: time indicator, map, and text box; we present this in Chapter 4. The "controller" includes how the user accesses, manipulates, and interacts with the data managed by the model and displayed through the view; we discuss this in Chapter 5. We also demonstrate how a user would be able to utilize the GNI to acquire Situation Awareness (SA) of for some missions using real data in Chapter 6.

It is convenient to re-list the five requirements from the previous chapter:

1. The GNI should have a chat-like text console to display auto-generated narrative summaries.
2. The GNI should have the capability to generate narratives at various granularities.
3. The GNI should have a map and a time indicator to handle spatiotemporal information.
4. The GNI should maintain cohesion between components described in requirements 1 and 2.
5. The GNI should process and display data in real-time.

2.1 Three Implementation Concepts

The GNI is an instantiation of three concepts that we derived from a combination of literature on narratives and literature on building model-based summaries in user interface design: *anytime summarization*, *storytelling*, and *multi-perspective analysis*. We discuss each concept here. We will explain how these concepts are related to the model-view-controller architecture in the following sections.

2.1.1 Anytime Summarization

Anytime summarization is a concept proposed by Shreckenghost [45] as a useful way to support a human operator who is interacting with remote robots, such as the mission described in Section 1.1. Anytime summarization provides users a provisional summary, perhaps in the form of mission-relevant metrics, of an in-progress mission [44]. Because of the provisional nature of the summary, it can be obtained at "any time." In terms of a narrative, an *anytime summarization embodies the metaphor of a plot summary.* Summaries are required according to Requirements 1 and 2 from the list, and they need to be created in real-time to satisfy Requirement 5.

In this thesis, summaries are created by analyzing the progress of a mission by comparing the mission plan to data obtained under actual execution. One main contribution of this thesis is the creation and evaluation of a set of algorithms that perform and summarize this comparison. This comparison may seem simple; however, there are many things to consider. There are various tasks to be performed in each mission, and for each task there are multiple categories of goals to be achieved. For example, if a task is to reach a certain point, the plan for the task should specify when the robot needs to be at the point and provide specific information about the location. There may be additional requirements like places to avoid on the way to the designated location or a required approach vector when navigating to a location. We compare each criterion like time, location, and other restrictions for the plan and the actual execution data to evaluate the degree of achievement of tasks. From the comparisons and associated evaluations, we extract a summary of how well a mission was accomplished.

2.1.2 Storytelling

There are numerous ways to tell a story with information; the proper way to tell the story, and hence to display the information, depends on (a) who is accessing the information, (b) what tasks that person must perform, and (c) the way that the information is encoded. We propose that the human supervisor's need for Situation Awareness (SA) determines how a story should be told.

Endsley has proposed that situation awareness, a human's sense of what is going on in the world, consists of three elements [15, emphasis added]:

> [Situation awareness is] the *perception* of the elements in the environment within a volume of time and space, the *comprehension* of their meaning, and the *projection* of their status in the near future.

Hone *et al.* explained that this situation awareness can be obtained by a human user through three steps: 1) as a grasp of a little segment of the environment (Transitory Awareness or TA), 2) as an understanding of causal relationships or connections between some segments of the environment (Local Awareness or LA), and 3) as a full understanding of the entire environment (Global Awareness or GA) [22]. TA can be obtained by examining data sent from a robot; this is what traditional GUIs tend to emphasize. LA can be obtained by reading abstracted summaries. Highly abstracted summaries highlight the key events of a mission and emphasize causes and their effects. GA can be obtained by studying detailed summaries and all other types of information previously stated.

Various information presentation methods are used by the state-of-the-art GUIs in an attempt to increase a user's SA. The problem of using lists, static images, and logical relations in a GUI is that these interface elements emphasize the perception of elements but not comprehension or projection. These conventional GUI elements are forms of *one-way communication*; the system throws information at a user, and it is the user's responsibility to give meaning and context to what is given. In order to increase the proficiency of the system, the user interface should allow not only more interactions between the data and the user, but also provide a mission-based context for comprehending information and projecting information into the future. To achieve this, the GNI uses storytelling.

As we stated in Chapter 1, storytelling includes "temporal ordering" and the manifestation of the connections of the events [59, 58]. This means that the generated narratives should not be simple lists of events, but rather series of some meaningful events that are related to each other to help human users to understand the entire missions and the explanation of the relationships between the events. To achieve this, the GNI should find the relationship and importance of events through analysis. For example, A and B are independent events. A simple list of event $\{A\ B\}$ may simply be interpreted as A then B. However, there are many different meanings that can be added to this like $\{A$ causes B to fail$\}$, $\{B$ is totally independent from $A\}$, or $\{B$ is a "redo" of $A\}$.

2.1.3 Multi-Perspective Analysis

Multi-perspective analysis, as the name indicates, analyzes stories from different perspectives. Humans can view the same thing from different perspectives depending on various conditions. When we analyze the data, we should do so from various perspectives so that we can capture the right context.

The first aspect of multi-perspective analysis is to change the scope of story [34]. This means how much data should be taken into the consideration when narratives are generated. If someone asks for some information about a movie, it does not always mean that the person would like to know about the entire movie. If the first half of the movie is boring, the information provided may be focused on the latter half of it. By narrowing down the scope of information, new "points of interest" that would be buried by other "more interesting" if the entire information is considered as the scope would be found. For example, there are six events A, B, C, D, E, and F. Event E and F have the most significant impact on the achievement of the mission and B has some significance too. When a narrative summary is generated for the entire mission, the latter half of the mission gets more attention and buries B. However, if the scope is narrowed down only to the first half of the mission, B gets more attention since it is the most important event in the scope.

The second aspect of this is to change the granularity of a story. This means to control the amount of detail included in the narratives independent from the scope of analysis. Regardless of the scope being narrow or wide, still there are many choices of how many details should be included for each event and character.

Chapter 3

Model

The phases used to create an anytime summarization in this thesis are: 1) represent transmitted packets as "data blocks", 2) store data blocks in a flexible database, 3) represent mission flow as a Kripke structure, 4) compare data blocks from the execution Kripke structure to the plan Kripke structure, 5) aggregate information from data block comparisons across a variable length sliding window to identify patterns and track causality, and 6) generating summaries at various granularities. These summaries are generated using automated data analysis algorithms at various granularities. This satisfies Requirement 2.

This chapter presents set of algorithms that perform and summarize this comparison. The chapter has a section for each of the five steps just listed. For example, Section 3.1 discusses the data block, Section 3.2 the database management of blocks, and so on.

3.1 Data Block

From the model-view-controller perspective, the model for summarization requires algorithms that integrate data received by the GNI over time. These algorithms are built on two data structures: a "block" structure and a graph structure.

The fundamental representation of data used in this implementation of the GNI is an implementation of a "block," a formatted chunk of information, as described in [34]. Figure 3.1 illustrates the structure of the block used in the GNI's data structure. A block is a snapshot of information received from the remote robot. Although there are many possible things that could go into a data block, this thesis assumes only two: telemetry and sensors.

As described in Chapter 1, the main elements of the mission require the robot to be at a particular place and time, so the data block in Figure 3.1 includes key telemetry information, namely temporal and geographical information. Additionally, the robot is equipped with sensors that are to be used at a particular time and place for a particular duration, so the status and result of sensor deployments are also included in a block. We explain each field in more detail in the next paragraph.

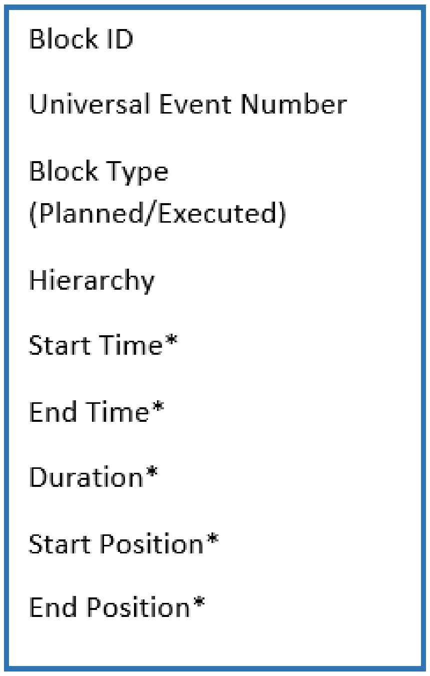


Figure 3.1: Structure of a block

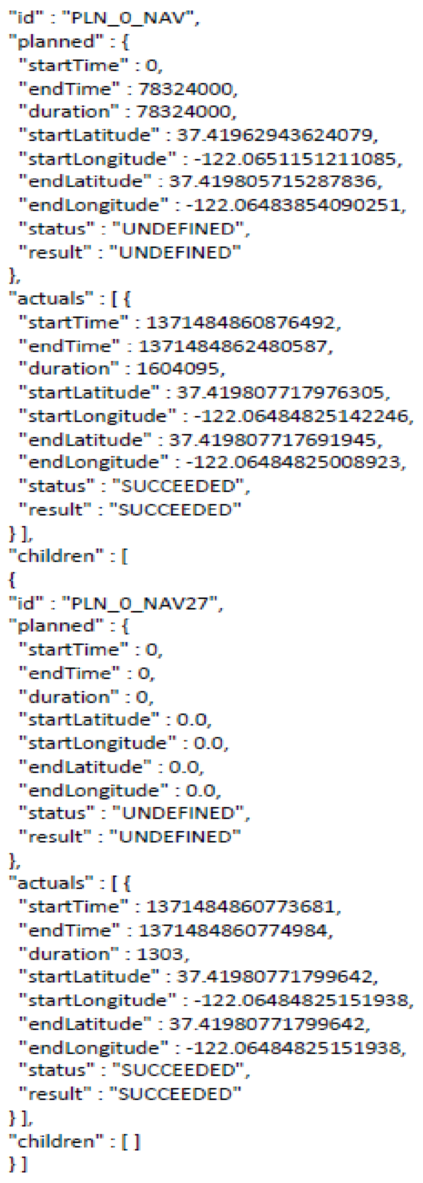
A block ID is included to coordinate blocks that are related to each other and to provide a unique ID that allows it to be tied to a telemetry block. A block ID consists of an event type (e.g. "move" meaning to travel to a certain location) and a number to represent the sequence of events that is defined in a mission plan within the same type of event (e.g. Move1). Besides the block ID, we put a universal number for each block to indicate the sequence of events within a mission. This number is independent from the number in the Block ID field (e.g. a block with a Block ID of "Move1" may have a Universal Event Number of "1", where another block with a Block ID of "Move2" may have a Universal Event Number of “5”). By using this universal event number, a user or programmer can quickly tell how many events are in a mission and at what point of a mission a specific event has occurred regardless of the event type.

The hierarchy field indicates whether the block has a parent and/or child block. This field is important because, in some cases, a task is separated into a sequence of sub tasks (e.g. multiple sensors need to be activated at the same location or a sensor needs to be activated while a robot is traveling to a destination). In these cases, each subtask becomes an independent block. To indicate the relationship between the blocks for a main task and subtasks, the hierarchy field is used.

Time is indicated by milliseconds and uses GMT as a time zone. Longitude and latitude values are used to indicate positions. An asterisk placed next to a field indicates that the field has two distinct values: a planned value and an actual value from the execution of a mission.

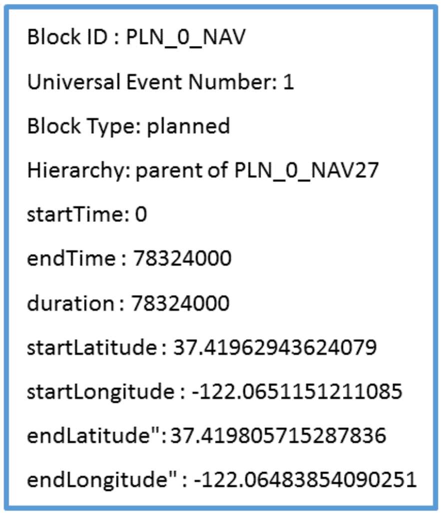
Conceptually, there could be many constraints about how a data block is transmitted like the communication bandwidth (i.e. the number of blocks than can be transferred at once) and the transmission order (i.e. which blocks should be sent out first) depending on the nature of a mission. For our project, we use data that are taken from already completed missions, so we assume there are no constraints to obtain blocks.

When we gather different types of data collected from different equipments of a UGV, we sort the data, group them by tasks, and put them into data blocks. These data blocks are transformed into a JSON file format. The GNI uses this JSON format file as a data source. The JSON format is a widely used format and pre-written parsers are available. The JSON format is easily read by both human users and computer systems. Figures 3.2 to 3.4 illustrate the first two tasks in the original JSON format and as individual data blocks.



```
"id" : "PLN_0_NAV",
"planned" : {
 "startTime" : 0,
 "endTime" : 78324000,
 "duration" : 78324000,
 "startLatitude" : 37.41962943624079,
 "startLongitude" : -122.0651151211085,
 "endLatitude" : 37.419805715287836,
 "endLongitude" : -122.06483854090251,
 "status" : "UNDEFINED",
 "result" : "UNDEFINED"
},
"actuals" : [ {
 "startTime" : 1371484860876492,
 "endTime" : 1371484862480587,
 "duration" : 1604095,
 "startLatitude" : 37.419807717976305,
 "startLongitude" : -122.06484825142246,
 "endLatitude" : 37.419807717691945,
 "endLongitude" : -122.06484825008923,
 "status" : "SUCCEEDED",
 "result" : "SUCCEEDED"
} ],
"children" : [
{
"id" : "PLN_0_NAV27",
"planned" : {
 "startTime" : 0,
 "endTime" : 0,
 "duration" : 0,
 "startLatitude" : 0.0,
 "startLongitude" : 0.0,
 "endLatitude" : 0.0,
 "endLongitude" : 0.0,
 "status" : "UNDEFINED",
 "result" : "UNDEFINED"
},
"actuals" : [ {
 "startTime" : 1371484860773681,
 "endTime" : 1371484860774984,
 "duration" : 1303,
 "startLatitude" : 37.41980771799642,
 "startLongitude" : -122.06484825151938,
 "endLatitude" : 37.41980771799642,
 "endLongitude" : -122.06484825151938,
 "status" : "SUCCEEDED",
 "result" : "SUCCEEDED"
} ],
"children" : [ ]
} ]
```

Figure 3.2: First two tasks in JSON format



Block ID : PLN_0_NAV

Universal Event Number: 1

Block Type: planned

Hierarchy: parent of PLN_0_NAV27

startTime: 0

endTime : 78324000

duration : 78324000

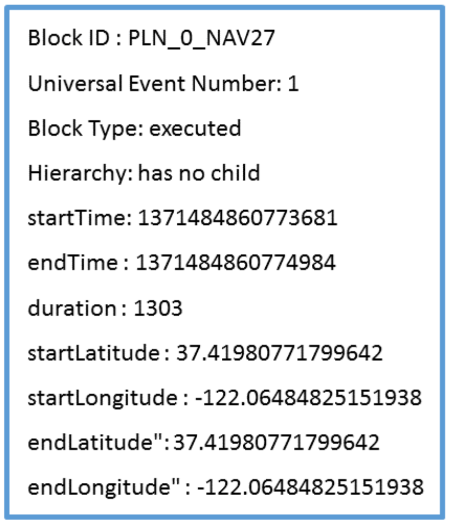
startLatitude : 37.41962943624079

startLongitude : -122.0651151211085

endLatitude":37.419805715287836

endLongitude" : -122.06483854090251

Figure 3.3: First Planned Data Block



Block ID : PLN_0_NAV27

Universal Event Number: 1

Block Type: executed

Hierarchy: has no child

startTime: 1371484860773681

endTime : 1371484860774984

duration : 1303

startLatitude : 37.41980771799642

startLongitude : -122.06484825151938

endLatitude":37.41980771799642

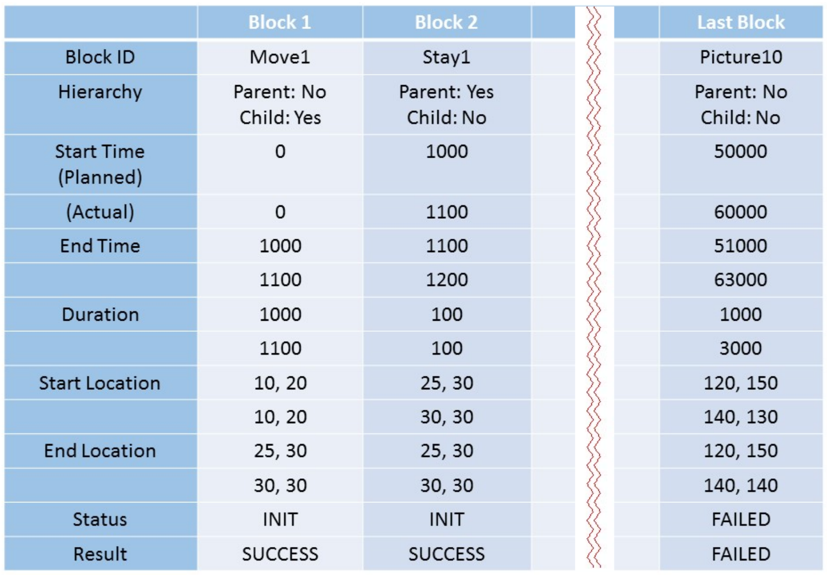
endLongitude" : -122.06484825151938

Figure 3.4: Second Executed Data Block

3.2 Block Database

When the GNI receives data blocks for a mission, whether a single block or multiple blocks at a time, a data table is created/updated to store all the blocks, as shown in Figure 3.5. This is the second step from the list given in the introduction to Section 3. All the data for a mission are included in a single JSON file and we load the file into our GNI.

If we were going to display data on the GNI in real time, we would need to find a way to manage block generation efficiently. If communication bandwidth between a robot and a ground control were limited, not all the data could be sent together. We leave these data transmission problems for the future work; however, we measure the time that the GNI takes to load a JSON file to generate a data table, analyze data, and generate narrative summaries to ensure the usability of this interface.



	Block 1	Block 2			Last Block
Block ID	Move1	Stay1			Picture10
Hierarchy	Parent: No Child: Yes	Parent: Yes Child: No			Parent: No Child: No
Start Time (Planned)	0	1000			50000
(Actual)	0	1100			60000
End Time	1000	1100			51000
	1100	1200			63000
Duration	1000	100			1000
	1100	100			3000
Start Location	10, 20	25, 30			120, 150
	10, 20	30, 30			140, 130
End Location	25, 30	25, 30			120, 150
	30, 30	30, 30			140, 140
Status	INIT	INIT			FAILED
Result	SUCCESS	SUCCESS			FAILED

Figure 3.5: Example of the data table

3.3 Representing Mission Flow

The third step from the list given in the introduction to Chapter 3 is to create a representation of a mission plan and an executed mission that can be used to analyze cause and effect.

We use a Kripke structure [1] to model the data for a mission [3]. A Kripke structure is a tuple $M = (S, I, T, \ell)$ with a finite set of states S, the set of initial states $I \subseteq S$, a transition relation between states $T \subseteq S \times S$, and the labeling of the states $\ell : S \to P(A)$ with atomic propositions A [3]. An example, adapted from [2], is illustrated in Figure 3.6. For the model represented in the example, $S = \{pay, select, soda, juice\}$, $I = \{pay\}$,
$T = \{(pay, select), (select, soda), (select, juice), (soda, pay), (juice, pay)\}$,
$\ell = \{(pay, \{\}), (select, \{paid\}), (soda, \{paid, drink\}), (juice, \{paid, drink\})\}$ and
$A = \{paid, drink\}$. For this example, transition conditions from one state to another are labeled as actions such as $Act = \{insert\_coin, get\_soda, get\_juice\}$

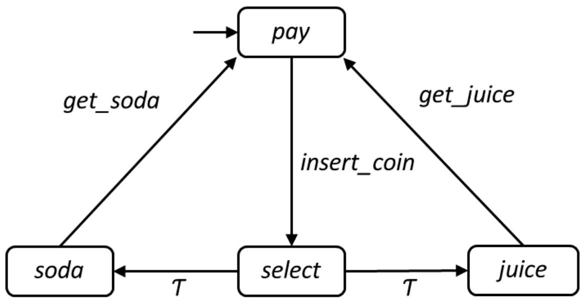


Figure 3.6: A transition system of a simple beverage vending machine.

For the model in our project, $S = \{$all valid combinations of values for data types specified for a data block (e.g. BlockID is a combination of task type and sequence number)$\}$, $I = \{$where the robot is placed at the beginning of a mission$\}$, $T = \{$list of valid rules for transition or conditions to meet to move onto the next event$\}$, $\ell = \{$what consists of a label (e.g. task type and an identification number)$\}$ and $A = \{$combinations of values for each task$\}$. For our project, transition conditions from one task to another are labeled as actions such as $Act = \{$the preconditions for the next task$\}$.

We now describe how we represent the *planned data* as a Kripke structure. We use each data block within a mission plan as a state in a Kripke structure. Information described

[1]A Kripke structure is a generalization of a finite state machine [7]. Some of the examples in this section can be implemented as finite state machines, but the algorithms we have written apply to more general relations.

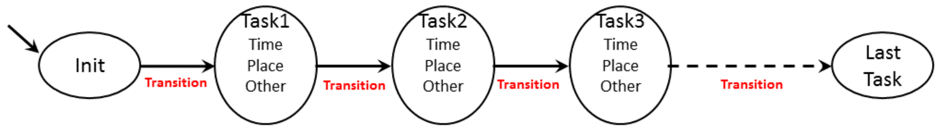


Figure 3.7: A mission plan represented in Kripke Structure.

in Figure 3.1 such as temporal, spatial, and other constraints for each task are used as the transition conditions from one state to the next as illustrated in Figure 3.7. These transition conditions are represented as propositional/predicate logic statements given the plan and constraints. Transition from a task to another can be completed when all the post conditions for the previous task are satisfied and also the preconditions for the following task are satisfied. More precisely, given a pair of sequential blocks it is possible for each and every data field to change between the blocks. The transition condition is the propositional logic statement that completely and exclusively specifies which transitions must change and which cannot.

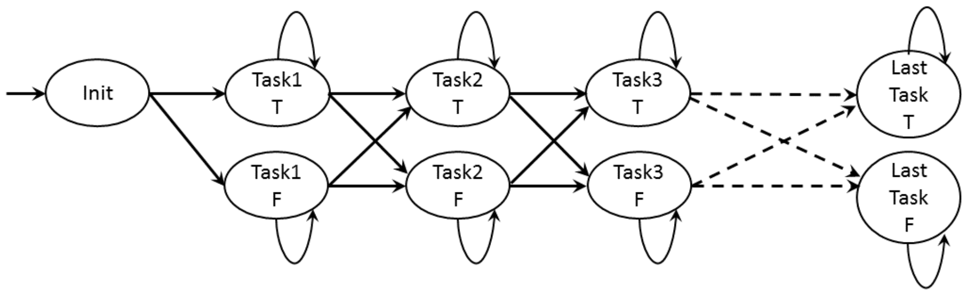


Figure 3.8: Possible execution data (single constraint or atomic proposition only) represented in Kripke Structure.

Next, we use the data blocks obtained from the *actual execution* of the mission and convert those into another Kripke structure as illustrated in Figure 3.8. Recall that it is feasible for each and every data field in adjacent blocks to change, and the transition condition between blocks in a plan is the conjunction of what must change and what must not change. By contrast, the transitions in an execution must account for every possible way that a plan can go wrong. We represent this in the execution Kripke structure by focusing on a single constraint or atomic proposition at a time. This means that we select one of the data fields from the block shown in Figure 3.1 and ignore all other fields.

For each task in a mission plan, there will usually be multiple constraints like time and location, and it is unlikely that a real robot in a real environment will meet all the requirements without any deviation. Therefore, the possible execution structure takes each possible change in a data field and creates two successor states that changes only one field at a time. One successor is a “success” successor, and the other is a “failure” successor. Thus, the possible execution structure has a path from the planned state to every possible next state. For each data field, success or failure depends on a predetermined tolerance. In equation 3.1, this tolerance is called a threshold.

The total number of possible execution traces is n^m where n is the number of valid outcomes for a single constraint of a task. If this is binary chose, being whether “good” or “bad”, then $n = 2$. m is the number of total tasks in a mission. We do not need to generate/expand all the traces at once, since there is only one actual execution trace according to the data. Thus, as we compare and evaluate each task constraint, we can generate/expand the path through the Kripke structure.

3.4 Comparing *Planned* to *Executed*

The central part of the summary generation is the analysis of data stored in the data table described in 3.2. Thus, this is related to both the anytime summarization and the storytelling concepts. There are two groups of data: “planned” and “executed”. The users want to know how well the robots executed the mission. To do this there are two types of data comparison: 1) comparison and analysis of the “planned” and “executed” data for each event or task in a mission, and 2) comparison and analysis of data for different events within the same data group (i.e., time, location, or other constraints) between multiple data blocks. The first comparison is necessary to measure the degree of accomplishment for each task. The second comparison is also important to understand the flow of the mission or to find causality of errors. We discuss further about the second type of comparison in Section 3.5.

This is an example for the first type of comparison. There is a data field called *Start Time*, and it has a single value in microseconds. The difference between time values for *planned* and *executed* should be either greater than or less than a threshold value of α. This threshold value represents a tolerance level of the values for each data field in a data block. Therefore, for *executed* data blocks, there should be two successive states from a previous state for a single constraint or atomic proposition. If the difference between an actual time value and a planned time value is less than a tolerance value, then we can consider that a task is performed well with regard to the aspect of time.

Now, we present an algorithm to perform these types of comparison. Let the symbol B denote a set of data blocks. Data blocks that contain the planned values can be represented as follows: $B_{\text{plan}} = \{\text{blocks} \mid \text{Block Type} = \text{“Planned”}\}$. $b_{\text{plan}} \in B_{\text{plan}}$ and $b_{\text{exec}} \in B_{\text{exec}}$ represent a single data block for each category. We can put a number after a block to indicate its Universal Event Number (e.g. $b_{\text{plan}}1$ is the data block that contains the planned values for the first event). Since each data block has multiple data fields, we use a dot after a block symbol to represent a value for a specific data field (e.g. $b_{\text{plan}}1.\textit{Start Time}$ represents a planned start time value of the first event in a mission).

We use an evaluation function of E to compare a planned data block and an executed data block and evaluate whether an event is successfully completed or not. The function is defined: $E_{\text{dataField}} : B_{\text{plan}}N.dataField \times B_{\text{exec}}N.dataField \rightarrow \{T, F\}$. We apply the function in Equation 3.1 to all the data fields for all the data blocks selected to analyze and evaluate the completeness of the selected events. The evaluation function for a *planned* and *execution* comparison is

$$E_{\text{dataField}}(b_{\text{plan}}N.dataField, b_{\text{exec}}N.dataField, \alpha) = \begin{cases} T & \text{if } |b_{\text{plan}}N.dataField - \\ & b_{\text{exec}}N.dataField| \leq \alpha^{\text{dataField}} \\ F & \text{otherwise} \end{cases} \tag{3.1}$$

where N stands for Universal Event Number, *dataField* stands for a data field of a data block, and $\alpha^{\text{dataField}}$ is a predefined threshold value for the specified data field.

We used a binary output in the evaluation function for simplicity, but it does not always have to be that way. We can separate the outcomes into multiple ranges and assign a score for each range of values. In the previous example of time, for example, a score of 1 can be assigned to the deviation range between 0 to 51 time units, a score of 2 to the range between 51 to a 100, and a score of 3 to the range 101 or more. In this example, receiving a lower score is better, meaning that the mission plan is accomplished more precisely.

3.5 Aggregating Information

An execution trace is the path taken through the possible execution Kripke structure associated with one transition from one planned state to another planned state. It can be represented as a string of true and false values.

Step 4 from the previous section gave a method for deciding how well a transition from a planned state to a planned successor state occurred. In this section, we address step 5, which performs two types of information aggregation: grading execution between a planned event and its successor event, and evaluating execution traces. We begin with the former.

3.5.1 Grading Execution Between An Event And Its Successor

The evaluation function from Equation (3.1) evaluates the degree of achievement for a single task. We modify the function to evaluate whether the execution result is getting closer to what has been planned or not between two events for each data type in a data block. This means that instead of evaluating whether the difference between a planned and execution values is less than a threshold value, we can compare the time difference between two consecutive events.

The evaluation function for a *execution* and *execution* comparison is

$$E_{\text{dataField}}(b_{\text{exec}}N.\mathit{dataFieldDiff}, b_{\text{exec}}M.\mathit{dataFieldDiff}) = \begin{cases} T & \text{if } b_{\text{exec}}N.\mathit{dataFieldDiff} - \\ & b_{\text{exec}}M.\mathit{dataFieldDiff} \\ F & \text{otherwise} \end{cases} \tag{3.2}$$

where M and N stand for Universal Event Number, $M \neq N$, $b_{\text{exec}}N$ is a successor of $b_{\text{exec}}M$, and *dataFieldDiff* stands for the difference of *planned* and *execution* values for a data field of a data block.

3.5.2 Aggregating Across an Execution Trace

The evaluation functions in Equations (3.1)-(3.2) evaluate the degree of achievement for a single or two consecutive tasks. It is possible to aggregate across multiple planned events. For simplicity, we only perform this type of aggregation within a common data type over an execution trace. We apply the evaluation functions of E that are defined in the previous section to all the chosen events and concatenate the output letters (i.e. T or F) to obtain output strings as shown in Equation (3.3).

$$OutputString = S(n) = S(n-1) \circ E_{\text{dataField}}(b_{\text{plan}}N.dataField, b_{\text{exec}}N.dataField, \alpha) \tag{3.3}$$

In the equation, n indicates the number of data block analyzed and the operator "$\circ$" indicates letter concatenation.

3.5.3 Discussion of the Aggregation Algorithm

Simple comparison between a planned data and an execution data for a single task is not enough to analyze the completion level of an entire mission, since tasks are not always independent from each other. Tasks are related in many different ways: types of task and order of those tasks are good example. The “move” task by itself is not very important since transferring a UGV from one place to another is usually not a main objective of a mission; more important are the different tasks performed by the UVG once it reaches a specific location or possibly on the way to a specific point. If there is a task of “take a photograph at location 1”, there should be a “move” task to location 1 as well. If the “photograph taking task” is canceled for some reason, the “move to location 1” task should also be canceled and be replaced by another “move” task to a next location.

When there is a delay in one task during a mission, it usually affects the subsequent tasks as well. However, depending of the type of the task that causes a delay, we can estimate different outcomes for later parts of the mission. When there is a delay in “take a photograph” task, we should suspect if the camera is working properly or not. If the same problem happens to multiple consecutive “photo taking” tasks, it might happen again later. So, the “photo taking” task might be canceled for the rest of the mission. However, even though there is a delay in completing the photo taking task, if the task is caused by the previous “move to a location” task, the “photo taking” task should not be canceled since it is not the main reason of the delay and the camera may be working fine.

By aggregating the information, we can understand and analyze the flow of a mission. This flow includes the causality of events and trends of whether the completion level is getting better or worse.

3.6 Generating Summaries

Step 6 is to generate abstractions from the aggregations that have been created in the previous steps. The capability of providing very detailed analysis would be appreciated by the user of the GNI in many cases, however, it may not be always the case. A large amount of information can overwhelm the user to find a specific piece information or make the user unable to grasp the flow of the entire mission. Therefore, we implemented a mechanism to reduce the amount of information depending on the user's needs. This abstraction method was designed to follow the principle suggested by Cook *et al.* [8], "One of the key characteristics of any summary is that it must be concise. To achieve this the content of the summary (1) must be focused on the key events, and (2) should leave out any information that the audience can infer on their own."

By concatenating the letters that are generated through individual comparisons, we can generate execution traces. We can analyze these execution traces and figure out the flow of the execution for the entire mission. However, these execution traces get longer when there are a large number of tasks in a mission and the number of traces also increases when there are larger numbers of data fields for a data block. Therefore, it is very important to be able to abstract the execution traces. There are two different ways to abstract execution traces. We present these abstraction methods and why they are so important.

The first abstraction method is to combine the execution traces for multiple data fields. We can do so by using the principle illustrated in Table 3.1. For this example, we limit the number of data fields in a data block to two for simplicity: time and location. As shown in Equation 3.1, there are only two outcomes: T or F. Therefore, there are only four possible outcomes if we are dealing with only two constraints. These four possible outcomes are labeled as "patterns" in Table 3.1. We assign a new output letter for each pattern to indicate different levels of completion for an event. By following this principle, a single output can represent the output for multiple constraints at once.

The second abstraction methods is to combine some symbols on an execution trace. We use the same principle that is illustrated in Table 3.1. An output string of 1-1-2-2-3 means there are five events/tasks analyzed with Equations (3.1)-(3.3) and according to the principles illustrated in Table 3.1. The first two events are completed perfectly (with respect

	Output for Time	Output for Location	New Output
Pattern 1	T	T	1
Pattern 2	T	F	2
Pattern 3	F	T	2
Pattern 4	F	F	3

Table 3.1: A table to combine outputs for two constraints into one.

to both *time* and *location* constraints). The middle two events have problems with either time or location. The last event has a problem with both. One way to abstract the sequence is to combine the same score if they are right next to each other. For example, the sequence of 1-1-2-2-3 would be abstracted into 1(2)-2(2)-3. The numbers in parentheses represent the number of scores combined. This indicates that the abstracted sequence has only three scores, 1-2-3, but there are still five events analyzed. This way of abstraction still captures the trend in the transition of the score; the tasks in the earlier stage of the mission were completed well, but the performance level dropped later on. By referring to the number of events/tasks merged together, which are represented by the numbers in parenthesis, the user can correctly determine at what point of the mission the performance change occurred. By combining some numbers (output letters) together, we can highlight the spots in a mission where the situation has changed dramatically. This is because, if we plot out all the output numbers as evaluation scores, we can find the places where the derivative of the plotted line becomes greater than zero. We summarize this concept and illustrate it in Figure 3.9[2].

3.6.1 Higher Levels of Abstraction

The user can change the degree of abstraction. When narrative summaries are requested by the user, the GNI must be able to generate abstracted information about the entire mission

[2]Another way to abstract the sequence is to use a special symbol for a specific pattern. For example, we can use the number 4 to replace a sequence of 1-2. We can also use 5 to replace 2-3. Then the original sequence of 1-1-2-2-3 would be 1-4-5. To use this method, there should be a lookup table that contains all the symbols used and their patterns of replacement. There should be more possible ways to abstract the output of comparison, but we leave those as out future work.

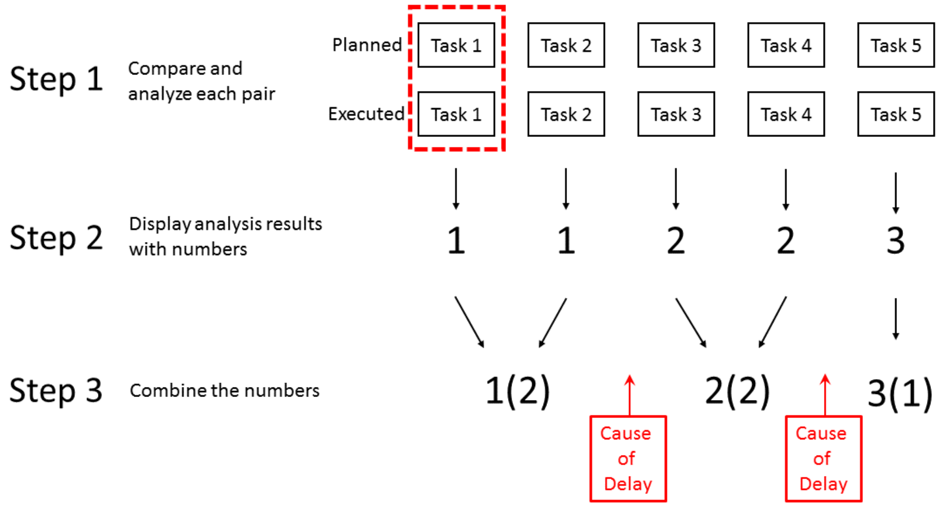


Figure 3.9: Summary of analyzing and representing data

that includes a minimum amount of details for each event. More detailed summaries should also be available to the user if he or she requests them. By allowing the user to request different detail levels, the GNI allows the user a simple way to control his or her perspective of the data —a simple instantiation of the multi-perspective presentation. We summarize the principles of information abstraction in Figure 3.10. The highest level of abstraction contains very brief information about the entire mission and, as the abstraction level is lowered, the summary contains more specific information for different parts of a mission. The volume of the summaries is relative to the details included in them: lower abstraction means contains more information and, therefore, more text information.

We present a text summary algorithm in Section 4.

3.6.2 Detecting Data Mismatch

We have discussed two types of data comparison. The first one was whether the "planned" and "executed" data match or not. The second one was about comparing the same data type to analyze the degree of progress. We now discuss about the third type of comparison to detect the "data mismatch". Data mismatch means there is a unusual sequence of data in

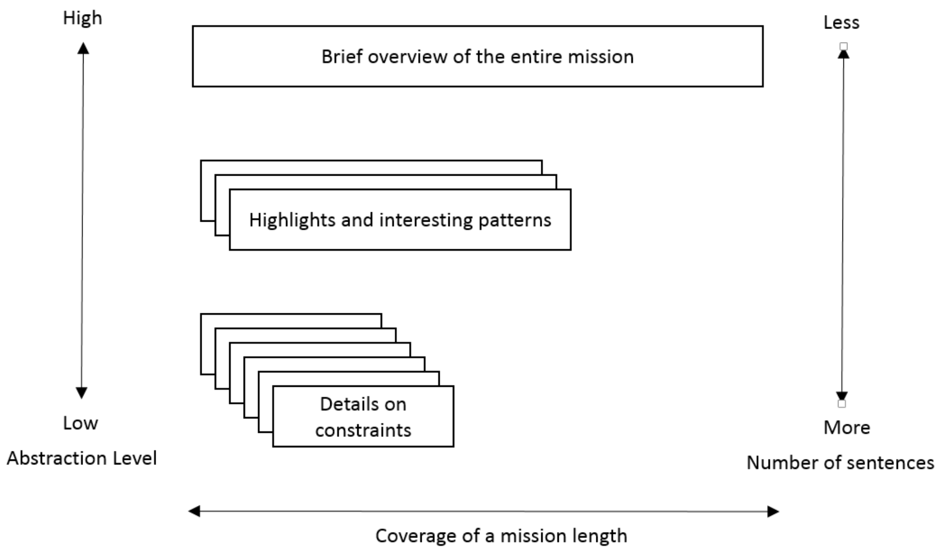

High
Low
Abstraction Level
Brief overview of the entire mission
Highlights and interesting patterns
Details on constraints
Less
More
Number of sentences
Coverage of a mission length

Figure 3.10: Overview of the principles of narrative generation and abstraction

either temporal or spatial data within “planned” or “executed” data type. For example, in our mission, a UGV traverses through way-points in order. Therefore, if a UGV visited three way-points “A”, “B”, and “C”, the end location for way-point A should be the same as the start position of way-point B. If this is not the case, we should suspect a data loss. The same principle can be applied to temporal data.

We can detect this type of mismatch by applying the principles of Equations (3.2) and (3.3). For example, to detect a location data mismatch for a plan data set, we use a modified version of Equation (3.2). The evaluation function for a *plan* and *plan* comparison is

$$E_{\text{location}}(b_{\text{plan}}N.endLocation, b_{\text{plan}}M.startLocation) = \begin{cases} T & \text{if } b_{\text{plan}}N.endLocation == \\ & b_{\text{plan}}M.startLocation \\ F & \text{otherwise} \end{cases} \tag{3.4}$$

where M and N stand for Universal Event Number and $b_{\text{plan}}N$ is a successor of $b_{\text{plan}}M$.

If Equation (3.4) returns F, it means that the transition from one event to its successor fails in Kripke structures and goes into "Data Mismatch" states as shown in Figures 3.11 and 3.12.

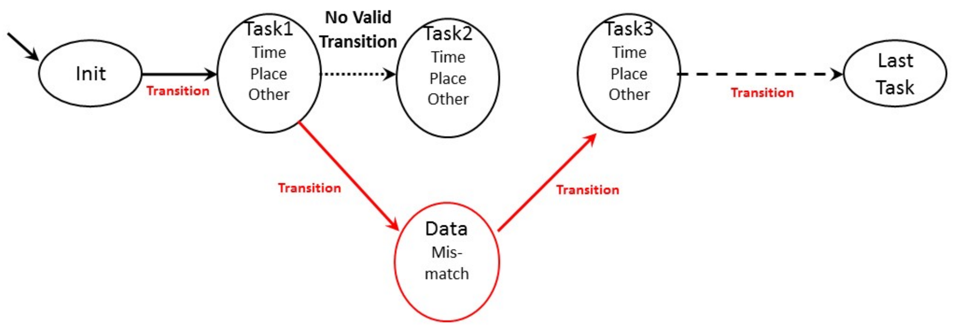


Figure 3.11: Data Mismatch on Plan Kripke Structure

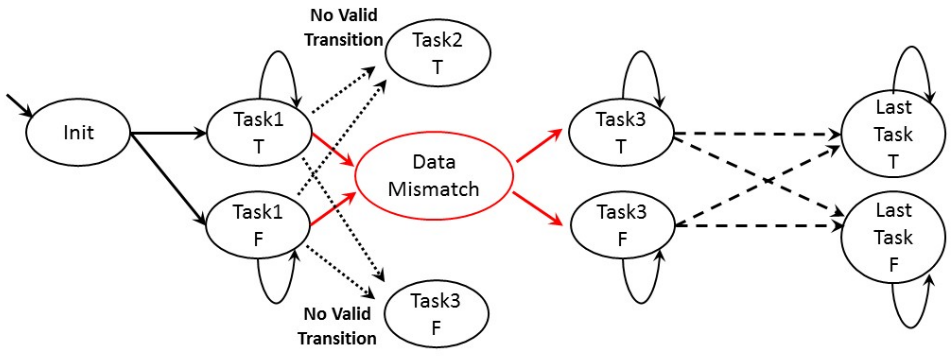


Figure 3.12: Data Mismatch on Possible Execution Kripke Structure

Chapter 4

View

Previous chapters have discussed how we model and manipulate the data. This chapter talks about how we present useful information to the user. We use a scheme of storytelling to communicate with the user through our interface.

Stories can be told through different types of information. Besides textual information, graphical aids can tell important important stories as well. These benefits might help human users to increase their situation awareness by helping them understand what is happening and why it is happening. Generated narrative summaries will be displayed when the button shown in Figure 4.1 is pressed.

4.1 Information Presentation

Traditionally, information is encoded as text, images, time series, geospatial information, or relations. As noted in Section 3.1, each of these information types is stored in the model in the form of raw data from the blocks and summaries from the aggregation algorithms.

There is considerable work on visually presenting information as text and images, so we select methods that align with the narrative metaphor. The primary relation that is relevant for the scenario presented in Chapter 1 is the relationship between the timestamp of a piece of information and the geospatial location for the source of that information, but other relations such as the relationship between a chat message and a location on a map are also useful.

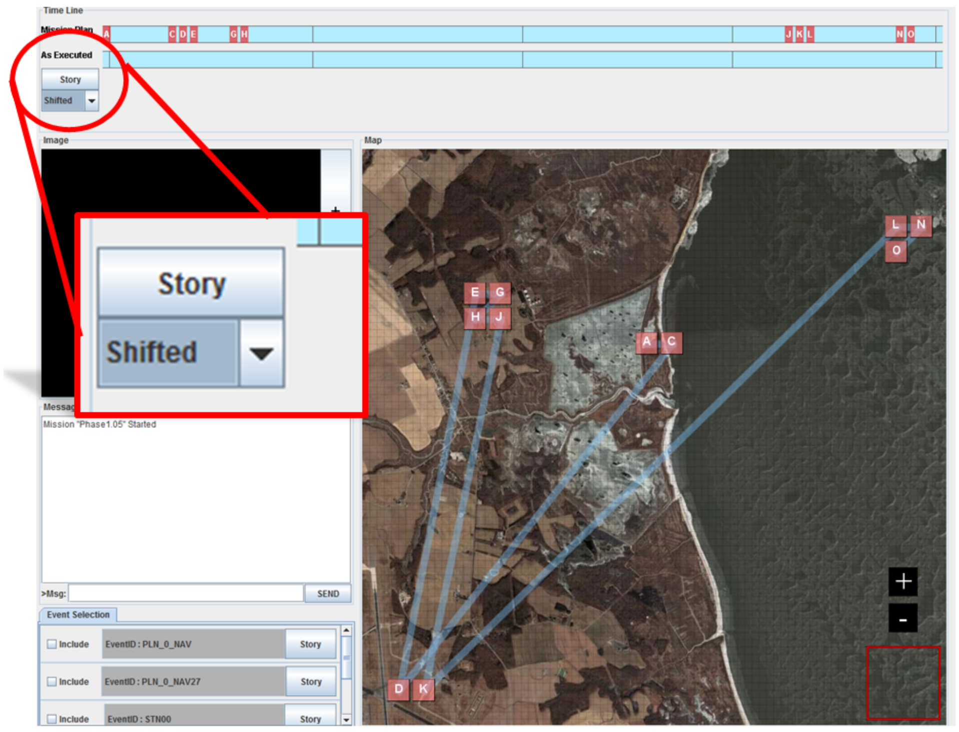
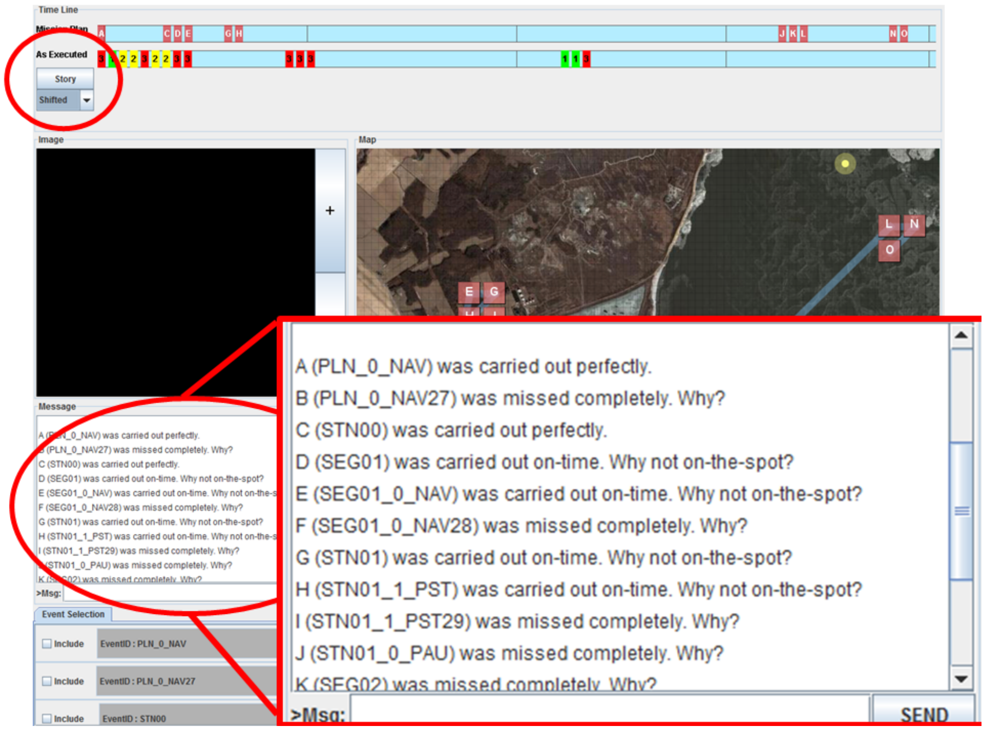


Figure 4.1: Story Generation Button

4.1.1 Text

The main part of storytelling on our interface is to use textual data. According to Matlock [30], certain vocabularies (verbs) or phrases let people create visual images in their minds. Also, Worth argues that "[there] are epistemological benefits to reading, hearing, and telling well-constructed narratives" [58]. This means that well-constructed narratives include a temporal ordering and cause-and-effect relationships. This helps readers to understand and to make sense of the storyline.

Using textual information like chat on a user interface has proven effective in real environments because users can "detect critical message contained in the data quickly and accurately" [6]. There are different types of textual formats, and each has its own merits and demerits. For information encoded in enumerated or bulleted lists of words, short phrases, or complete sentences, it is possible for a large amount of information to be displayed at once. However, each type also has its own demerits and, if misused, users may waste their time



A (PLN_0_NAV) was carried out perfectly.
B (PLN_0_NAV27) was missed completely. Why?
C (STN00) was carried out perfectly.
D (SEG01) was carried out on-time. Why not on-the-spot?
E (SEG01_0_NAV) was carried out on-time. Why not on-the-spot?
F (SEG01_0_NAV28) was missed completely. Why?
G (STN01) was carried out on-time. Why not on-the-spot?
H (STN01_1_PST) was carried out on-time. Why not on-the-spot?
I (STN01_1_PST29) was missed completely. Why?
J (STN01_0_PAU) was missed completely. Why?
K (SEG02) was missed completely. Why?
>Msg:

Figure 4.2: Enlarged Text Box

finding what they really want among the text without a clue of how it is organized. Lists and bullet point styles are handy to give a brief idea overall, but cannot portray a cause-and-effect relationships nor give specifics. Complete sentences are useful to give the precise context of the author, but on the other hand, because they give a lot of information, it may be more difficult to find some specific information that the audience wants quickly. Short phrases can have the merits and demerits of both lists and complete sentences depending on how they are generated. We propose to explore the use of a chat-based interface for presenting text-based information using various types of text presentation explored above.

For a high-level summary, a graph or chart may work effectively to present a few core messages from the data. In fact, Schreckenghost used charts and graphs when she introduced the concept of "anytime summarization" [45]. However, graphs and charts can be complicated and difficult to read if there is too much information presented in them. Using many simple graphs and charts may take more space compared to plain text. For example, Demir et al. succeeded to generate textual summaries from any bar charts keeping the same amount of core messages contained in the charts in the same or smaller space [13]. We decided to use text to represent narrative summaries and we supplemented the text with timeline and map-based information. Future work should consider how to blend other types of data summaries and graphs with text, timelines, and map-based information.

There are many different ways to generate text with computer programs. However, as discussed in [41], there are mainly two groups of methods: using methods known as Natural Language Generation (NLG) and using templates. The NLG method, as the names implies, tries to imitate how a human writes natural language by understanding the semantics and syntactics of the language. The second method uses a limited number of templates already prepared. The task of the system is to choose the right templates and populate the templates with the suitable words. NLG methods have much higher capabilities to express various types of texts, but they are far more complicated to implement [54, 55].

Templates are less adaptive than NLG, but if a situation is known to the user, they can perform well enough. Therefore, templates are used much more often than NLG methods. Since the missions for which the GNI would be used are known to us, we use templates to generate our summaries. We chose templates that express temporal and spatial information.

The sentence structures are set up to fit the type of the situation where the interface would be used. So, the prepared sentences look like: "######<###>######". Sections with a symbol "#" are filled with words. Words between angled brackets (i.e. "<" and ">") are the variables of this grammar. We list a few examples. Event <EventDescription> happened at location <LocationDescription> at time <TimeDescription/SequenceDescription>. This is encoded as a grammar, so a key research element was to construct a simple grammar for a particular robot task.

In Section 3.6, we discussed about preparing multiple levels of abstraction to meet the needs of the users more efficiently. The suitable number of different abstraction levels depends on the number of blocks and the number of constraints given for each block. We currently use five different levels due to the number of events contained in a mission and the number of data fields defined for an event. At the highest level, the summary describes the entire mission. At the lowest level, the summary describes the details about each data field for each block by comparing the planned and the execution data. In the middle level, the connections between blocks and some tendencies or patterns for some constraints are highlighted. These principles are illustrated in Figure 3.10.

The prepared sentences for the highest level, or the most abstracted summaries are consisted of the templates that look like “Overall, the mission <succeeded/failed>”, “There are <numeric value > number of <planned/execution > tasks”. The prepared sentences for the lowest level, or the least abstracted summaries are consisted of the templates that look like “The <planned/actual> value for task <eventID> is <real data>”. The prepared sentences for the lowest level, or the least abstracted summaries are consisted of the templates that look like “The <planned/actual> value for task <eventID> is <real data>”.

By populating the templates of sentences with variable information obtained from the analysis, we form and present stories[1]. This satisfies Requirement 1.

4.1.2 Images

Some missions include a task to capture images at certain locations. Therefore, we have an image display. There is an example provided as Figure 4.3. The image display has a zooming capability to show more details. Use buttons marked as No. 1 in the figure. Users can check the level of zooming by reading a gage marked as No. 4. If an original image size is bigger than the display size, the user can move the image displayed area by directly clicking on the image. The display area is separated into four areas with respect to the cursor displayed on the center of the console (marked as No. 3), and when the user clicks one of the direction (up, down, left, or right), the displayed area moves according to the user input. Users can check which region of the original image is displayed by the gage marked as No. 2.

[1]We are able to define the templates for the sentence and define a number of abstraction levels because the context of missions that we would be dealing with is known to us. If this information is not accessible, then we need to have a more generalized method to analyze the input data, define templates of sentences for narrative summaries, and define a suitable number of abstraction levels. This is future work.

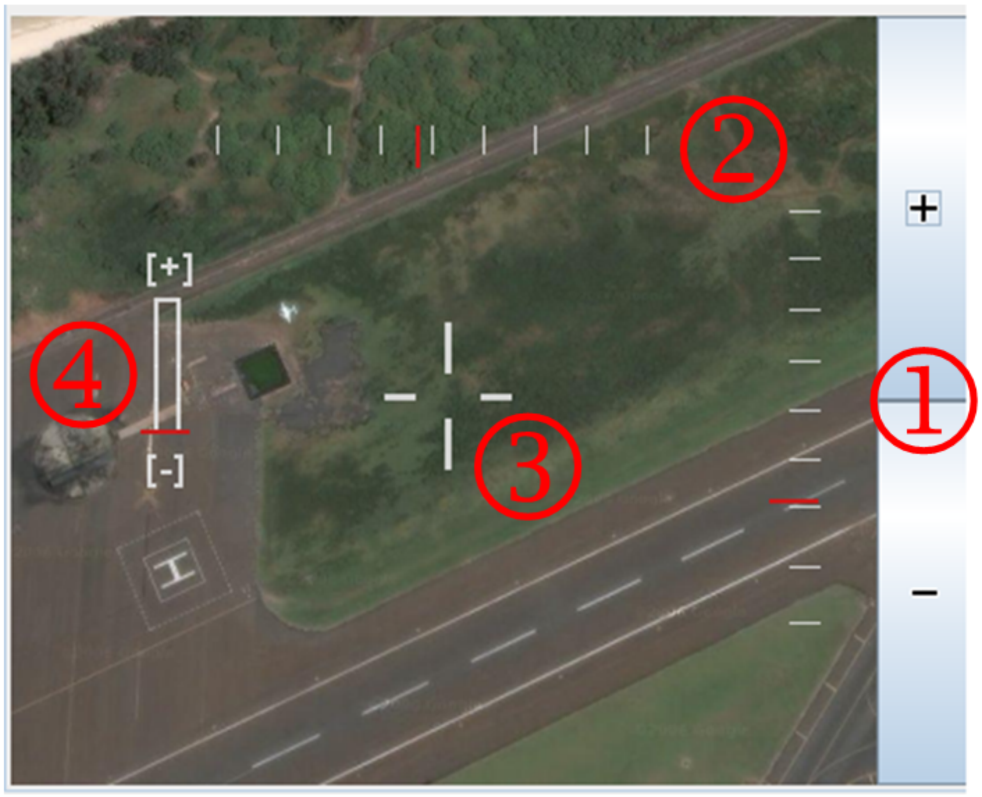
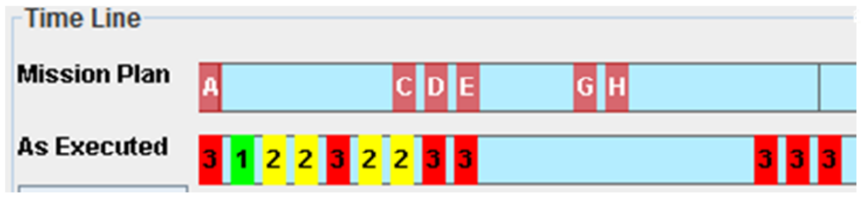


Figure 4.3: Example of the Image Display Console

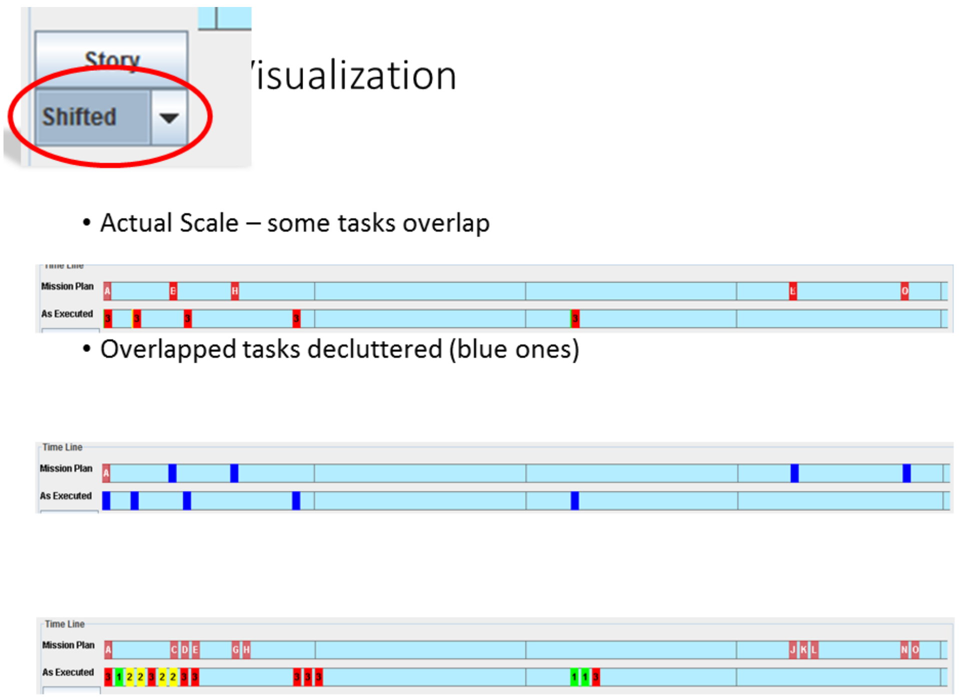
4.1.3 Time Series



- Green with 1: Both time and location values are all in the right range
- Yellow with 2: Either time or location value is in the right range
- Red with 3: Neither value is in the right range

Figure 4.4: Explanation of the icons used on the Time Indicator

This component represents the time flow of the entire mission from initiation to termination. There are two distinct time lines placed one over the other, each representing a mission plan and actual execution, to be compared easily. When a task icon is clicked on either a plan or execution timeline, the corresponding event icon on the other timeline would



• Actual Scale – some tasks overlap

• Overlapped tasks decluttered (blue ones)

Figure 4.5: Time Indicator Magnification Example

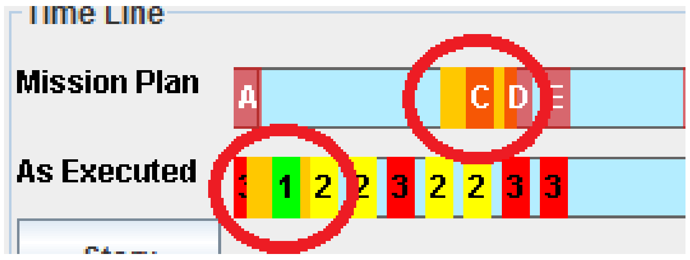


Figure 4.6: Linked icons on Time Indicators.

be highlighted to indicate the relation as shown in Figure 4.6. We adapted the techniques from Park's time indicator [39] to fit with the other elements of the GNI. We implemented 1) the use of symbols to indicate details of events and 2) zooming.

This indicator uses multiple colors, numbers and letters, as shown in Figure 4.4. Each event icon shown on the mission plan timeline has an alphabet letter as its ID. Each event icon shown on the execution timeline has one of three colors: green, yellow, or red. The color represents the grade given to the spatial aspect of the execution trace. If a task is performed close enough to the planned point, the icon is colored in green. If a task is performed not close but not too far either, then the icon is colored in yellow. If a task is performed completely out-of-place, then the icon is colored in red. Each event icon on the execution timeline also has a number between 1 to 3, indicating the grade given to the timing aspect of the execution trace. 1 means close to perfection, 3 means otherwise, and 2 means somewhere in between those two. By using different symbols, the human user can grasp the overview of the mission quickly.

One of the major problems that Park dealt with in using a time indicator is it overwhelms the user with too much information or unreadably small fonts around the indicator. To solve this, he recommended using a zooming function that reveals or hides different levels of detail. To achieve this, we display events on the timeline in three different layouts: 1) true to the original time scale, 2) near-by tasks combined, and 3) shifted as shown in Figure 4.5. Layout 1 spreads events out according to the true proportion of the starting time of the events. The left edge of the timeline is the starting time of the mission and the right edge means the end of the mission. For layout 1, events are placed based on the actual scale of the time. Layout 2 combines event icons when they are grouped closely together. Those "combined" icons are colored blue to be distinguished from single event icons easily. In layout 3, if events would appear too close to each other then subsequent event icons are shifted to the right just enough to show their ID. Therefore, the placement of the event icons are a little bit off from the actual scaling, but each icon's ID are clearly shown. By switching between different display modes, the user should be able to know the intervals of the events true to the actual time scale and still be able to see the details about each event.

Another feature related to zooming is pop-up windows which display detailed information about each event. When a user right clicks on an event icon, a small pop-up window will appear to show details about the event. This helps satisfy Requirement 3.

4.1.4 Geospatial Information

Figure 4.7 shows a snapshot of the geospatial map used in the GNI. We use a simple two-dimentional aerial map for two reasons: 1) aerial maps are easily understood by human users [40], and 2) because we use a ground vehicle, we do not need to have detailed information about elevation.

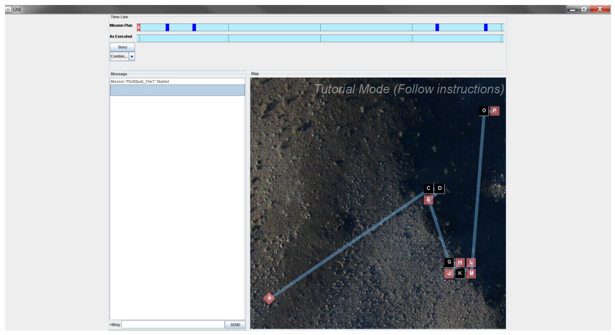


Figure 4.7: Screen Shot of a Map of the GNI.

In our project, the main usage of a map is to display the distribution and grades of points of interest over a certain area. A two-dimensional map gives the user enough information to understand the terrain and help to interpret the correlation between the points specified.

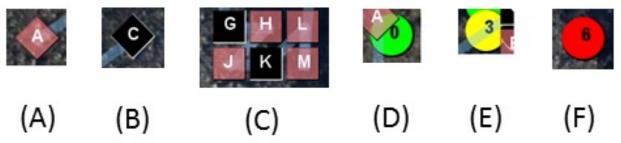


Figure 4.8: List of the Icons Used on the Map.

There are two groups of icons used on the map as shown in Figure 4.8. First group contains plan icons (i.e. icons marked as (A), (B), and (C)) and second group contains execution icons (i.e. icons marked as (D), (E), and (F)). The icon marked as (A) is a plan icon. The letter embeded on the center of the icon is the ID. The icon marked as (B) is also a plan icon, but it has a different color from the previous one. It is because this type of plan icon does not have any corresponding execution icon. The icons marked as (C) are a mixture of two different kinds of plan icons, but they have a different shape compared to the previous ones. It is because these icons are grouped together since their original locations are too close to each other and their icons overlap each other. Since these icons are grouped together, their current locations on the map are slightly deviated from their original locations.

Icons marked as (D), (E), and (F) are execution icons. The shape and the size are identical. The differences are the color and the number embeded on the center of icons. The number indicates the number of data fields in the data block for which this event received a good or poor grade. For example, if there are total of five data fields in a data block and a task receives "T" for three different fields using Equation 3.1, the number this icon shows is "2". Since this number indicates the number of data fields a task "failed" to complete, smaller numbers mean more successful completion of a task. The color indicates the same principle. For the numbers of missed data fields less than three, a color of green is assigned. For the numbers between three to five, a color of yellow is assigned. For the number greater than five, a color of red is assigned. Execution icons do not change their shapes even when they are grouped together.

By using different shape, color, and letter, users can understand different types of information represented by these icons easily. This satisfies Requirement 3.

Chapter 5

Controller

The previous two chapters discussed the model (data manipulation) and the view (data display). This chapter discusses the last element of implementation, which is the controller. This means how a human user can acquire desired information from the interface. In this section, we discuss two features used to allow the human to control this portion of the model: *event selector* and *relations*.

When the GNI displays information, the user can respond to it through various actions like selecting events, comparing icons on different components, or requesting more detailed information. For example, right-clicking on an icon on any component will pop up a new small window with detailed information about the event. This supports two-way communication between the GNI and the user.

5.1 Event Selector

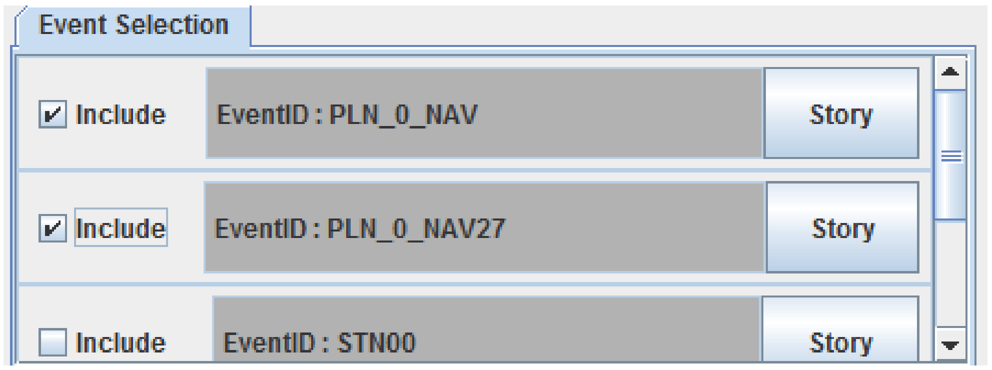


Figure 5.1: Enlarged Event Selector

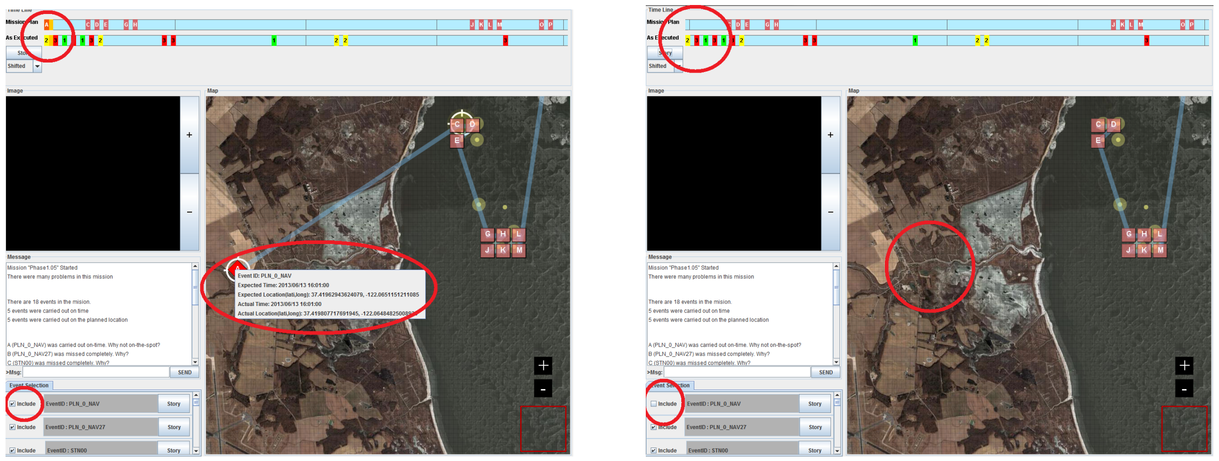
The idea of the event selector is related to the key feature of multi-perspective analysis. Instead of displaying all sorts of information, a user will pick whatever type of information is desired for the narrative summary. As shown in Figure 5.1, the event selector is a list of all the events in a mission. The one on the top is the first event. There is a check box on the left side of each event icon. If the box is checked, the event is taken into consideration when narrative summaries are generated. Through this feature, users can include only the necessary events into the summaries.

For example, if a user selects only a few events from the top of the list on the event selector, generated narratives are only focused on the first few events of a mission and only the selected events' icons are displayed on each component of the GNI. Another example is to select only a specific type of event. There are usually multiple types of tasks to be performed in a mission and the eventID displays the type of task. By selecting a single type of task or a few related types of tasks, a more focused analysis can be obtained.

We demonstrate how to use the event selector and the result is illustrated in Figure 5.2. For this demonstration, we check or un-check the "include" box for the first event on the event selector. When the box is checked, we can see the icons for the first event on both timeline and map. However, when the box is un-checked, those icons are not displayed. Only the events with the "include" boxed checked would be displayed on the components of the GNI.

5.2 Relations

Each component of the GNI is designed to tell narratives on its own. Thus, users can obtain very specific narratives by focusing on a single component. This helps the user to gain Transitory Awareness (TA) as discussed in 2.1.2. However, by referring to all the components and their narrative summaries, the user can obtain an understanding of the entire mission, and thus, can move toward Global Awareness (GA).



Mission "Phase1.05" Started
There were many problems in this mission

There are 18 events in the mision.
5 events were carried out on time
5 events were carried out on the planned location

A (PLN_0_NAV) was carried out on-time. Why not on-the-spot?
B (PLN_0_NAV27) was missed completely. Why?
C (STN00) was missed completely. Why?

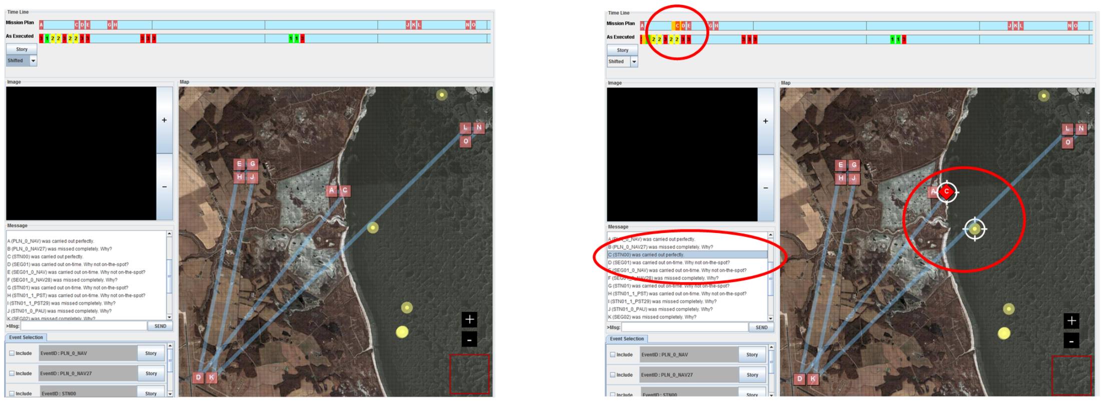
Event ID: PLN_0_NAV
Expected Time: 2013/06/13 16:01:00
Expected Location(lati,long): 37.41962943624079, -122.0651151211085
Actual Time: 2013/06/13 16:01:00
Actual Location(lati,long): 37.419807717691945, -122.0648482500893

✓ Include	EventID : PLN_0_NAV	Story
✓ Include	EventID : PLN_0_NAV27	Story
✓ Include	EventID : STN00	Story

(a) With Check (b) Without Check

Figure 5.2: Use of Event Selector

We implemented two features in the GNI to increase the users' accessibility to the desired information on the interface. The first feature is a hyperlink-like connection between icons and areas in different GNI components. As illustrated in Figure 5.3, when an icon or information about a task on the timeline, map, or text box is selected, all the icons or information that represent the same task in other components will be highlighted. Through this feature, users can distinguish which icons on different components are related and displaying the information of the same task easily.



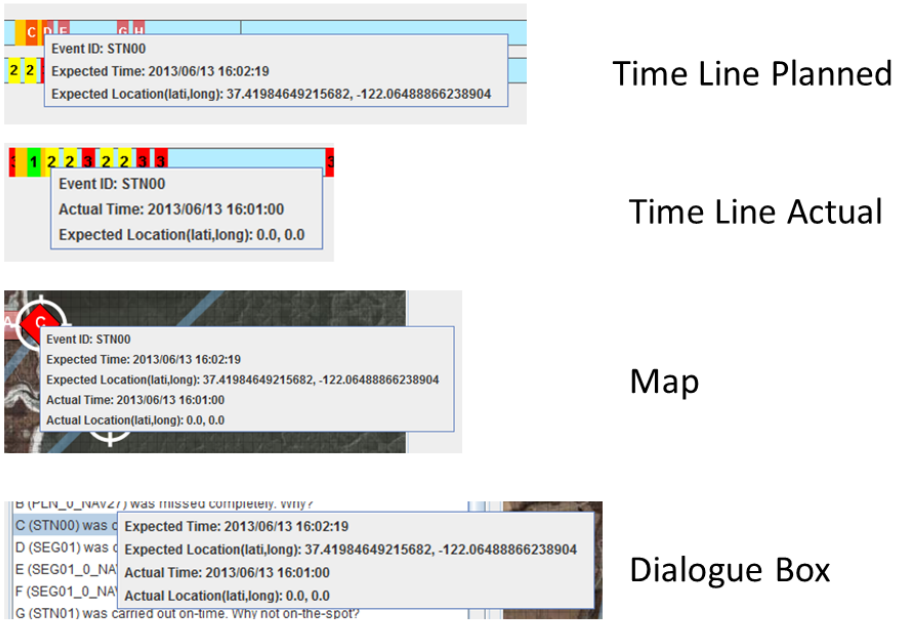
A (PLN_0_NAV) was carried out perfectly.
B (PLN_0_NAV27) was missed completely. Why?
C (STN00) was carried out perfectly.
D (SEG01) was carried out on-time. Why not on-the-spot?
E (SEG01_0_NAV) was carried out on-time. Why not on-the-spot?
F (SEG01_0_NAV28) was missed completely. Why?
G (STN01) was carried out on-time. Why not on-the-spot?
H (STN01_1_PST) was carried out on-time. Why not on-the-spot?
I (STN01_1_PST29) was missed completely. Why?
J (STN01_0_PAU) was missed completely. Why?
K (SEG02) was missed completely. Why?

Include	EventID : PLN_0_NAV	Story
Include	EventID : PLN_0_NAV27	Story
Include	EventID : STN00	Story

(a) Before (b) After

Figure 5.3: Before and After of Mouse Click

Second, by using the right click of the mouse, detailed information will be displayed in a pop-up window as illustrated in Figure 5.4. Regardless of the component the clicked icon is displayed, the user can access any kind of detailed information through the pop-up window. The temporal information displayed on the timeline are processed separately and independently from the spatial information displayed on the map. However, since all the information is stored on the same data table introduced in 3.5, we can find all the related information about a task when a information request is made from any component on the GNI.



Event ID: STN00
Expected Time: 2013/06/13 16:02:19
Expected Location(lati,long): 37.41984649215682, -122.06488866238904

Event ID: STN00
Actual Time: 2013/06/13 16:01:00
Expected Location(lati,long): 0.0, 0.0

Event ID: STN00
Expected Time: 2013/06/13 16:02:19
Expected Location(lati,long): 37.41984649215682, -122.06488866238904
Actual Time: 2013/06/13 16:01:00
Actual Location(lati,long): 0.0, 0.0

B (PLN_0_NAV27) was missed completely. Why?
C (STN00) was c
D (SEG01) was c
E (SEG01_0_NA
F (SEG01_0_NA
G (STN01) was carried out on-time. Why not on-the-spot?

Expected Time: 2013/06/13 16:02:19
Expected Location(lati,long): 37.41984649215682, -122.06488866238904
Actual Time: 2013/06/13 16:01:00
Actual Location(lati,long): 0.0, 0.0

Figure 5.4: Event Details View

Each component of the GNI tells different kinds of narratives. For example, by studying the information displayed on the timeline, users can understand the order of the tasks, global and relative spacing or timing of each event within a mission, and the difference between planned and executed time. However, users may not fully understand how these temporal narratives are related to spatial narratives. However, through these two features discussed above, users can find all the related information easily and quickly across different components of the GNI and it ties all the different types of narratives into one. It brings the unity to the interface. This satisfies Requirement 4.

Chapter 6

Demonstration of the Features of GNI

The thesis statement makes two claims. First, we claim that narrative summaries can be constructed from a mission plan and a history of mission execution. To satisfy this claim, we showed and explained about our narrative generation algorithms in Section 3. Second, we claim that the GNI can provide meaningful narrative summaries for a human supervisor in a human-robot team. To satisfy this claim, we showed and explained about each component of the GNI and how narratives would be displayed on those components. To support our two claims further, we review the five implementation requirements and how we satisfied those. Then, we simulate and demonstrate how a user would be able to utilize the GNI in a real situation by using an actual data set [5].

6.1 Implementation Requirements

For convenience, the five requirements of the implementation of the GNI again are listed below.

1. The GNI should have a chat-like text console to display auto-generated narrative summaries.
2. The GNI should have the capability to generate narratives at various granularities.
3. The GNI should have a map and a time indicator to handle spatiotemporal information.
4. The GNI should maintain cohesion between components described in requirements 1 and 2.

5. The GNI should process and display data in real-time.

6.1.1 Requirement 1

The basic concept of narrative generation is to analyze the data through various types of comparisons like comparison of *planned* and *execution* data and comparison between the same data type. Through these analysis processes, the GNI finds what information would be valuable to report for human users. This information includes the degree of achievement of a mission, situations where the performance of the robot changes, detection of data type mismatch, and detection of causality.

The degree of achievement means how many events satisfied desired conditions. In our projects, we have two main conditions for every event: temporal and spatial. Therefore, to report which event was executed on time and on the right spot is very important. Also, if there was any delay, it is useful to know which events caused delays and how those delays affected the rest of the mission.

6.1.2 Requirement 2

The GNI has a capability of generating narratives in different granularity levels. According to the number of information types and the total amount of data, this should vary. Currently, the number of levels has to be set by a human user manually. In our missions, we compare time and location values between planned and execution, and the total number of tasks is not so large (average number of tasks < 30). From these constraints, we decided that five levels would be enough to capture different levels of abstraction of our missions. The schemes of analysis are based on these pre-defined levels. We decided what information should be presented in each level and set a standard of how to analyze the data for each level.

When the GNI is launched, the narratives are displayed at level 1, meaning the most broad and general information that covers a wide range of a mission. Users can request narratives at different levels by pressing the story button. Figures 6.1 to 6.5 display the actual summaries generated for a short mission that consisted of four events.

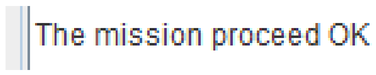

The mission proceed OK

Figure 6.1: Example of summary at level 1

Summaries at level 1 describe the entire mission in some simple phrases: either the mission went well, OK, or bad. “Well” means the majority of the events were executed successfully. “Bad” means the opposite: the majority of the events were executed poorly. “OK” is in the middle of “well” and “bad”: some events were executed well, but others not so well. Through these simple phrases, human users can grasp the general sense of achievement of the mission. Besides that, the users can tell what kind of information they expect as they further investigate the mission (e.g if a mission went “well”, the majority of the events should have been executed positively.) Thus, the user can focus on where the problems occurred easily.

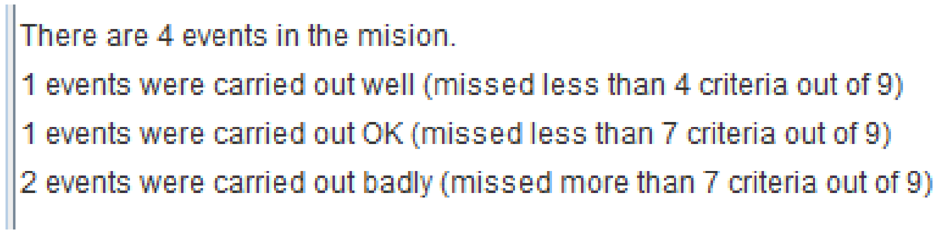
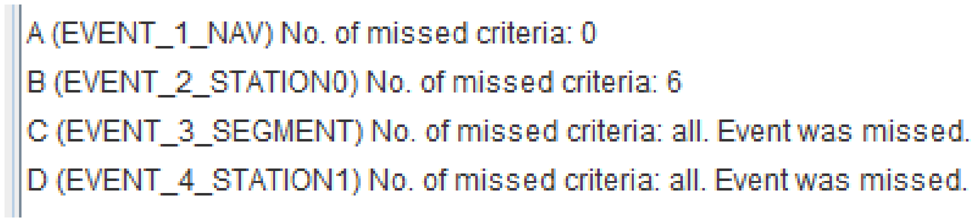

There are 4 events in the mision.
1 events were carried out well (missed less than 4 criteria out of 9)
1 events were carried out OK (missed less than 7 criteria out of 9)
2 events were carried out badly (missed more than 7 criteria out of 9)

Figure 6.2: Example of summary at level 2

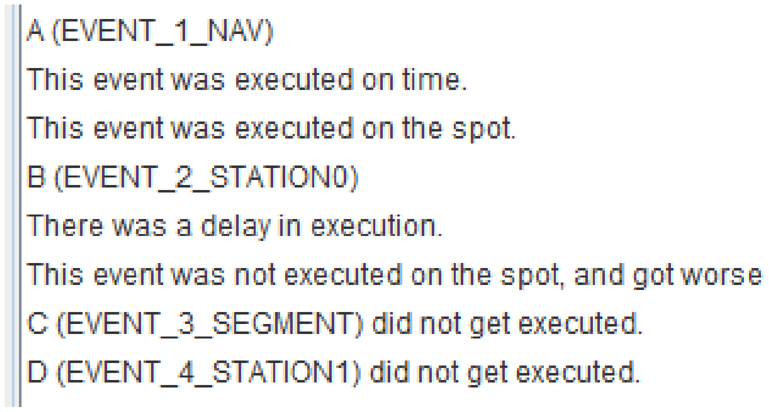
Summaries at level 2 report the actual number of events for each of the completion degrees (i.e. reports the total number of events for a mission and how many events are categorized as “well” and so on). In this level, displayed information is little bit more detailed than the previous one. Instead of describing the entire mission in a few words, summarized information is given as ratios. We do this not only for the general summary as whole mission, but also for each data type. If a mission is to analyze data collected by an automated rover as described in 1.1, then temporal information and spatial information are the main data types. So, summaries should include the information like following: Total event=10. General information: well=7, ok=1, bad=2. Temporal aspect: well=5, ok=3, bad=2. Spatial aspect: well=8, ok=0, bad=2.



A (EVENT_1_NAV) No. of missed criteria: 0
B (EVENT_2_STATION0) No. of missed criteria: 6
C (EVENT_3_SEGMENT) No. of missed criteria: all. Event was missed.
D (EVENT_4_STATION1) No. of missed criteria: all. Event was missed.

Figure 6.3: Example of summary at level 3

Summaries at level 3 give information about a few specific parts of a mission that have higher importance compared to the rest. This means the GNI is required to identify some key events in a mission. For example, if a serious problem happened and the mission was terminated, it should be captured as a key event. A key event may consist of multiple events since it is highly likely that events are related to each other in some ways. Causal relationships are a good example; a small error in a previous event may cause greater effects later.

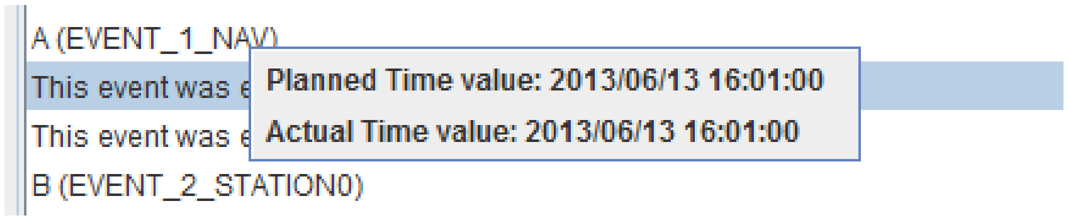


A (EVENT_1_NAV)
This event was executed on time.
This event was executed on the spot.
B (EVENT_2_STATION0)
There was a delay in execution.
This event was not executed on the spot, and got worse
C (EVENT_3_SEGMENT) did not get executed.
D (EVENT_4_STATION1) did not get executed.

Figure 6.4: Example of summary at level 4

Summaries at level 4 report each event. At this level, the main focus is how each event was performed. Usually there are multiple conditions to be satisfied for each event. This level displays how those conditions were met.

Summaries at level 5 display the raw data. It means to display actual "planned" and "execution" data for each event. This is the most detailed information, and at the same time,


A (EVENT_1_NAV)

This event was e

This event was e

B (EVENT_2_STATION0)

Planned Time value: 2013/06/13 16:01:00

Actual Time value: 2013/06/13 16:01:00

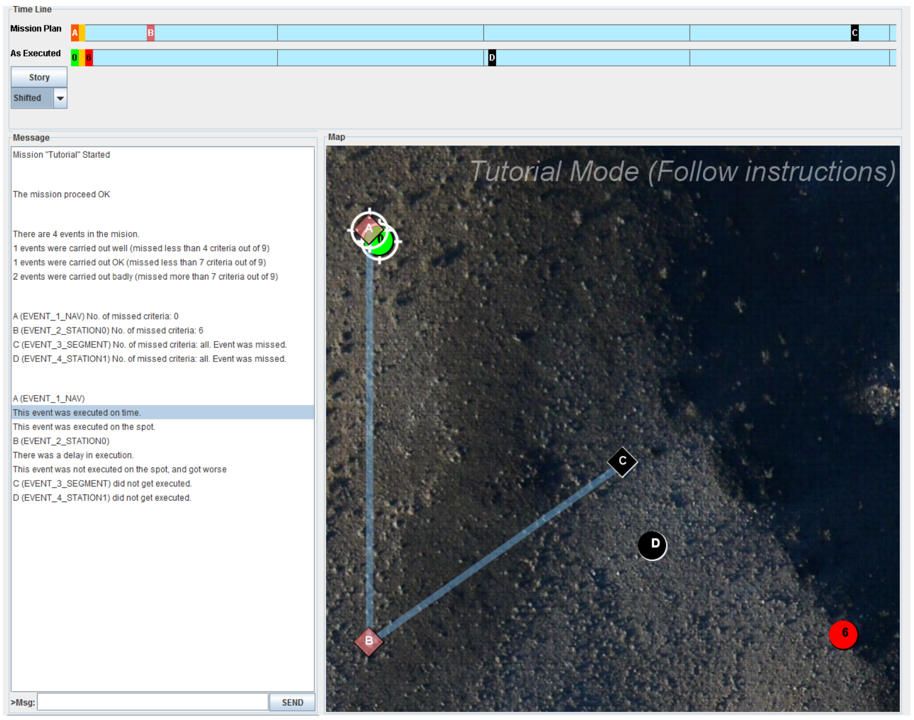
Figure 6.5: Example of summary at level 5

this is the most unorganized data. The users must make sense and understand the meaning behind these data.

Many traditional GUIs are focused to display information at level 5. This kind of information can help users to gain LA, but not much TA or GA. The GNI, on the other hand, not only provides the level 5 information but also information from level 1 to level 4 in order. We think the order is very important. The GNI starts by providing s small portion of information that is easy to understand. Then, it adds more details. This avoids overwhelming the users with a large volume of information at once. Also, when very detailed information is presented, users should have gained enough understanding about the mission and be able to utilize the detailed information without confusion. If the information is provided with a reverse order, meaning the users first receive information at level 5 then proceed to receive level 1, users need to understand what all the detailed data mean and generate a story line in their minds. After this process, summaries will be provided. By this time, the user should have generated similar types of summaries in their minds trying to get TA and GA. Therefore, it would not help the users much. These additional levels presented in a right order would help users to gain better TA and GA.

6.1.3 Requirement 3

Figures 6.6 to 6.8 display execution of three missions of different durations. Because of the layout, shape, color, and the number embeded in the icons, the users may be able to surmize the overall execution degree and identify the critical places and times of the mission.



Mission "Tutorial" Started

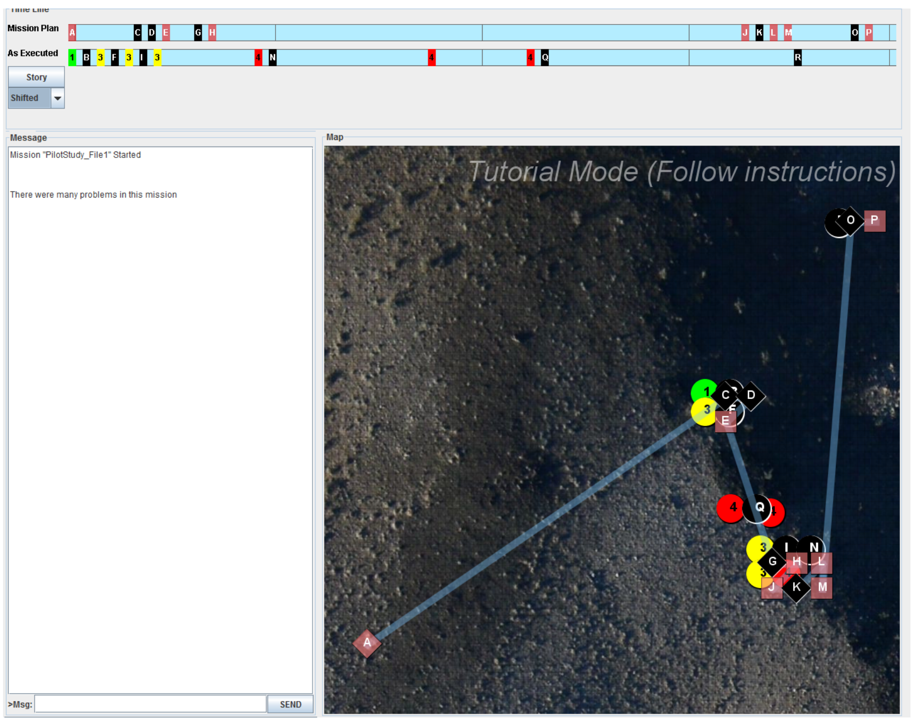
The mission proceed OK

There are 4 events in the mision.
1 events were carried out well (missed less than 4 criteria out of 9)
1 events were carried out OK (missed less than 7 criteria out of 9)
2 events were carried out badly (missed more than 7 criteria out of 9)

A (EVENT_1_NAV) No. of missed criteria: 0
B (EVENT_2_STATION0) No. of missed criteria: 6
C (EVENT_3_SEGMENT) No. of missed criteria: all. Event was missed.
D (EVENT_4_STATION1) No. of missed criteria: all. Event was missed.

A (EVENT_1_NAV)
This event was executed on time.
This event was executed on the spot.
B (EVENT_2_STATION0)
There was a delay in execution.
This event was not executed on the spot, and got worse
C (EVENT_3_SEGMENT) did not get executed.
D (EVENT_4_STATION1) did not get executed.

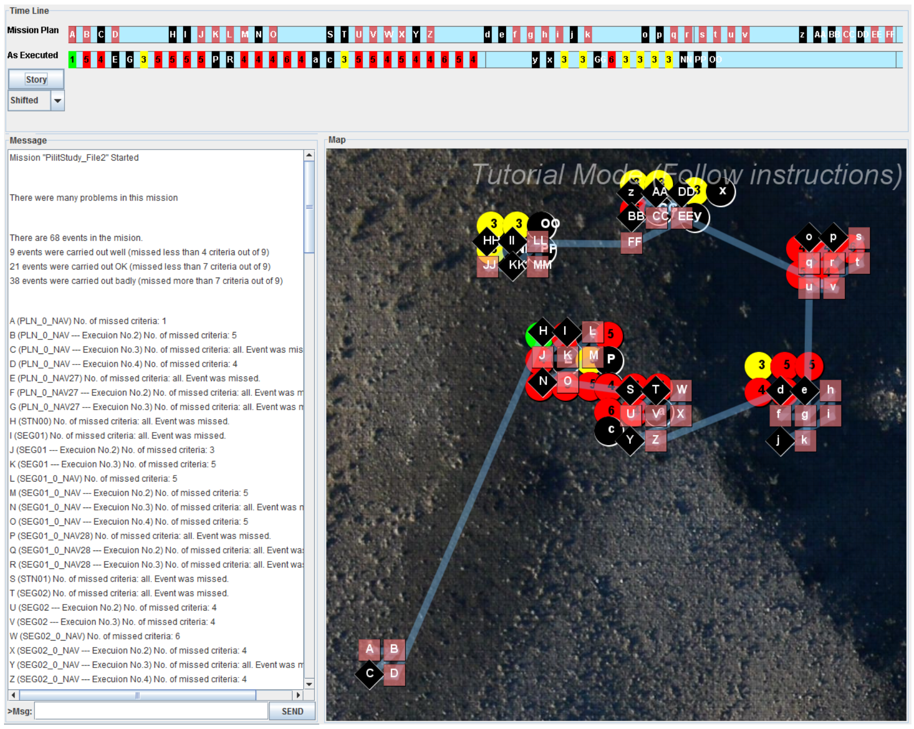
Figure 6.6: Example execution of a short mission



Mission "PilotStudy_File1" Started

There were many problems in this mission

Figure 6.7: Example execution of a mid-length mission



```
Mission "PilitStudy_File2" Started


There were many problems in this mission


There are 68 events in the mision.
9 events were carried out well (missed less than 4 criteria out of 9)
21 events were carried out OK (missed less than 7 criteria out of 9)
38 events were carried out badly (missed more than 7 criteria out of 9)


A (PLN_0_NAV) No. of missed criteria: 1
B (PLN_0_NAV --- Execuion No.2) No. of missed criteria: 5
C (PLN_0_NAV --- Execuion No.3) No. of missed criteria: all. Event was mis
D (PLN_0_NAV --- Execuion No.4) No. of missed criteria: 4
E (PLN_0_NAV27) No. of missed criteria: all. Event was missed.
F (PLN_0_NAV27 --- Execuion No.2) No. of missed criteria: all. Event was m
G (PLN_0_NAV27 --- Execuion No.3) No. of missed criteria: all. Event was m
H (STN00) No. of missed criteria: all. Event was missed.
I (SEG01) No. of missed criteria: all. Event was missed.
J (SEG01 --- Execuion No.2) No. of missed criteria: 3
K (SEG01 --- Execuion No.3) No. of missed criteria: 5
L (SEG01_0_NAV) No. of missed criteria: 5
M (SEG01_0_NAV --- Execuion No.2) No. of missed criteria: 5
N (SEG01_0_NAV --- Execuion No.3) No. of missed criteria: all. Event was m
O (SEG01_0_NAV --- Execuion No.4) No. of missed criteria: 5
P (SEG01_0_NAV28) No. of missed criteria: all. Event was missed.
Q (SEG01_0_NAV28 --- Execuion No.2) No. of missed criteria: all. Event wa
R (SEG01_0_NAV28 --- Execuion No.3) No. of missed criteria: all. Event wa
S (STN01) No. of missed criteria: all. Event was missed.
T (SEG02) No. of missed criteria: all. Event was missed.
U (SEG02 --- Execuion No.2) No. of missed criteria: 4
V (SEG02 --- Execuion No.3) No. of missed criteria: 4
W (SEG02_0_NAV) No. of missed criteria: 6
X (SEG02_0_NAV --- Execuion No.2) No. of missed criteria: 4
Y (SEG02_0_NAV --- Execuion No.3) No. of missed criteria: all. Event was m
Z (SEG02_0_NAV --- Execuion No.4) No. of missed criteria: 4
```

Figure 6.8: Example execution of a long mission

6.1.4 Requirement 4

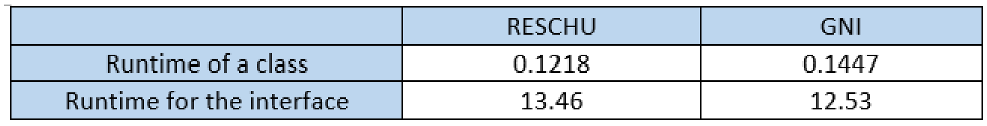
As displayed in Figure 5.3, icons on different components of the GNI are interlinked and when an icon is selected, all the related icons will be highlighted. Through this, the user can obtain very useful information of 1) the temporal gap between "planned" and "execution" data by finding corresponding icons on the timeline, 2) the spatial gap between "planned" and "execution" data by finding corresponding icons on the map, 3) the degree of completion of a task by the color of the icon, and 4)whether the event was executed or not by the number of highlighted icons.

6.1.5 Requirement 5

Processing time is an important factor in the concept of zation. The interface should be able to provide summarized data to the user whenever the user desires. Therefore, the difference in process time for the GNI and a GUI which does not provide the narrative summaries should be minimized.

The GNI prototype was built from an existing interface called RESCHU [1]. We compared the interface execution times in two different ways: 1) the run time of the specific class that includes narrative summary generation algorithms, and 2) the time required for the whole interface to be ready.

We measure time for each interface described above ten times and calculated the average in seconds. The results are shown in Table 6.1[1]. There was a little difference in runtime of a class, but it was not significant. RESCHU runs faster since it does not generate a data table, analyze it, or produce narrative summaries. On the other hand, the total runtime for GNI was shorter than RESCHU, because some of the RESCHU functionalities were disabled in the GNI. From this result, we can conclude that the narrative generation does not take too long to process; this satisfies Requirement 5.



	RESCHU	GNI
Runtime of a class	0.1218	0.1447
Runtime for the interface	13.46	12.53

Table 6.1: Runtime Comparison Results. Number unit is seconds.

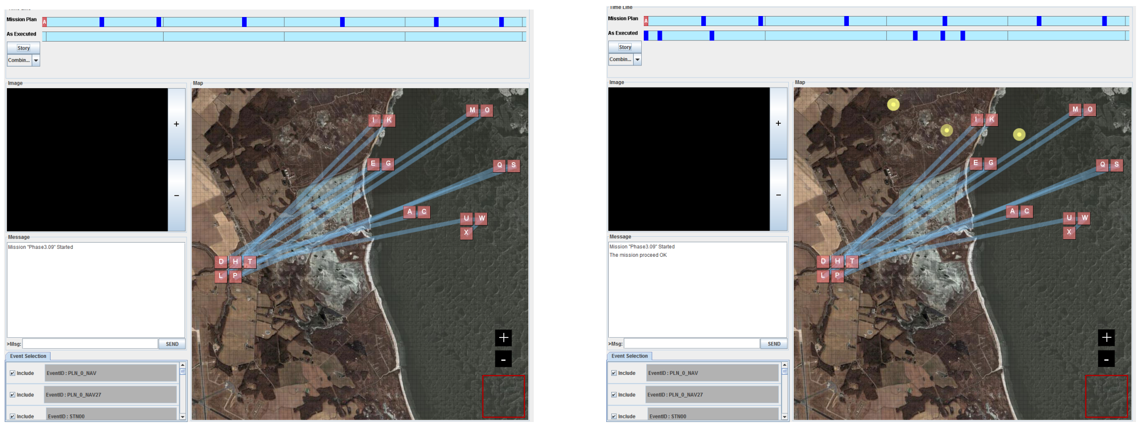
6.2 Conceptual Interaction

In this section, we present a conceptual interaction between the GNI and a user. Consider a GNI user who is both a mission manager and a scientist who wants to utilize the data obtained by the robot. The user uses the GNI to review a completed mission. Suppose that the objective of the mission is to explore a remote place by using an automated rover. The

[1]The machine used for this experiment is running on Microsoft Windows 7 64-bit. To measure the runtime of a class, we put a timer inside of the class. We started the timer at the beginning of the class and stopped at the end of the class. To measure the time for the interface to be launched, we put a timer in the main class of the interface. We started the timer before executing the main function to launch the interface and stopped it when the function closes.

rover has a pre-programmed plan to traverse through way-points. The data displayed in the figures are taken from one of the missions of the HET surface telerobotics mission [5].

When the user requests information about a mission, the GNI first presents a high-level summary of the entire mission on components 1 (timeline), 2 (map), and 3 (text box) in Figure 2.1. The before and after displays are illustrated as Figure 6.9. The user can request more detailed information on all or some selected components. The user also can select a certain amount of information using the component 4 (event selector) in Figure 2.1.



Mission "Phase3.09" Started

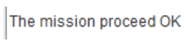
Mission "Phase3.09" Started
The mission proceed OK

(a) Before Dispalying Summaries (b) After

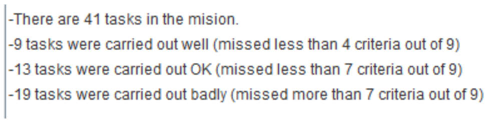
Figure 6.9: First Interaction

From a mission manager's perspective, the user may want to know the degree of mission completion. This includes how many way-points were successfully reached and how much time was spent on the entire mission. These kinds of general information are included in the narrative summaries so that the user may be able to grasp the overall achievement of a mission quickly (Figure 6.10). The user can request more detailed narratives to investigate if there were any minor problems such as short-term delays or unnecessary maneuvers.

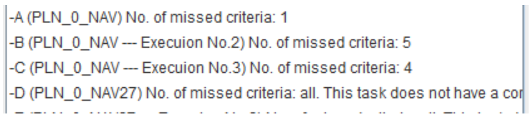
From a scientist's perspective, it is beneficial to know whether planned experiments were performed as they should or the reason for failure, if any. To do this, the user can use the event selector. This component lets the user select only necessary events and eliminate others. On Figure 6.11, only the "navigation" tasks are selected. Through this, the user can find the information of certain events quickly. Also, by clicking on an event icon, all


The mission proceed OK

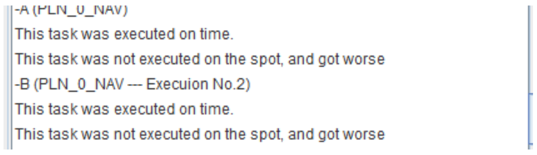
(a) Summaries at Lv.1


-There are 41 tasks in the mision.
-9 tasks were carried out well (missed less than 4 criteria out of 9)
-13 tasks were carried out OK (missed less than 7 criteria out of 9)
-19 tasks were carried out badly (missed more than 7 criteria out of 9)

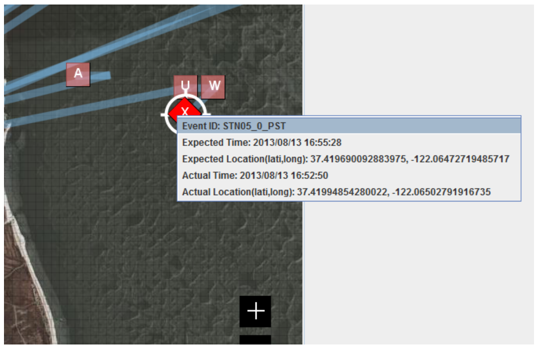
(b) Summaries at Lv.2


-A (PLN_0_NAV) No. of missed criteria: 1
-B (PLN_0_NAV --- Execuion No.2) No. of missed criteria: 5
-C (PLN_0_NAV --- Execuion No.3) No. of missed criteria: 4
-D (PLN_0_NAV27) No. of missed criteria: all. This task does not have a co

(c) Summaries at Lv.3


-A (PLN_0_NAV)
This task was executed on time.
This task was not executed on the spot, and got worse
-B (PLN_0_NAV --- Execuion No.2)
This task was executed on time.
This task was not executed on the spot, and got worse

(d) Summaries at Lv.4



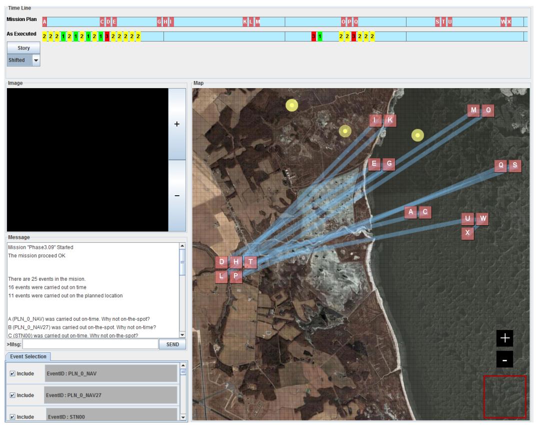
Event ID: STN05_0_PST
Expected Time: 2013/08/13 16:55:28
Expected Location(lati,long): 37.419690092883975, -122.06472719485717
Actual Time: 2013/08/13 16:52:50
Actual Location(lati,long): 37.41994854280022, -122.06502791916735

(e) Summaries at Lv.5 or right-clicking on an icon

Figure 6.10: Generated Summaries at Different Levels

other icons for the same event will be highlighted. On Figure 6.12, task "T" or "SEG02" is selected. You can see how all the related icons are highlighted. Through this, the user can easily tell the planned and actual locations of the event on the map and the planned and actual execution time of the event on the timeline. By right-clicking on an event icon, more detailed information can be obtained.

y



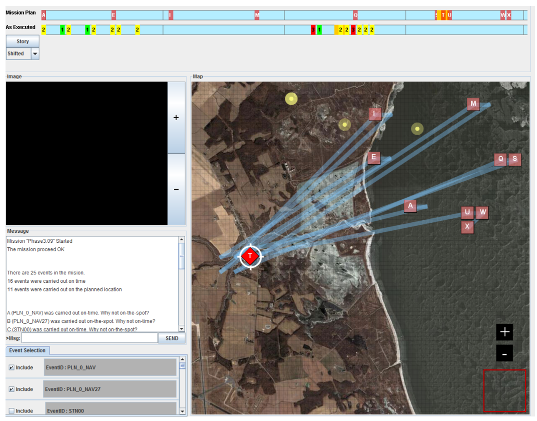
Mission "Phase3.09" Started
The mission proceed OK

There are 25 events in the mission.
16 events were carried out on time
11 events were carried out on the planned location

A (PLN_0_NAV) was carried out on-time. Why not on-the-spot?
B (PLN_0_NAV27) was carried out on-the-spot. Why not on-time?
C (STN00) was carried out on-time. Why not on-the-spot?

Include EventID : PLN_0_NAV
Include EventID : PLN_0_NAV27
Include EventID : STN00

(a) All the events displayed



Mission "Phase3.09" Started
The mission proceed OK

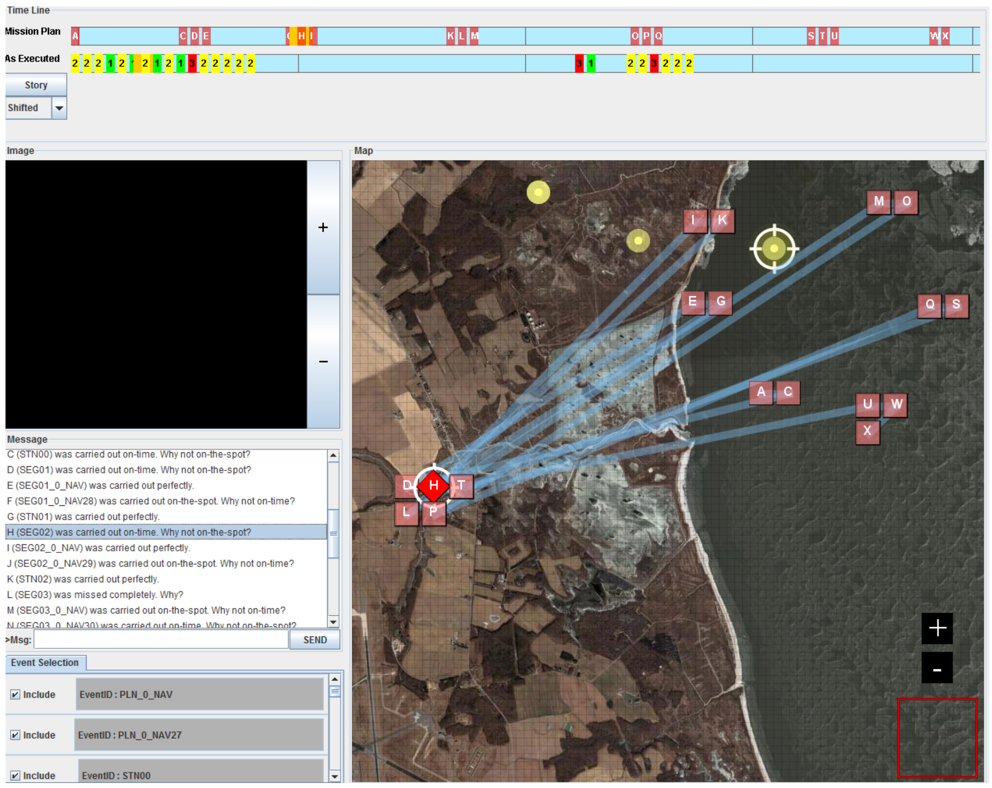
There are 25 events in the mission.
16 events were carried out on time
11 events were carried out on the planned location

A (PLN_0_NAV) was carried out on-time. Why not on-the-spot?
B (PLN_0_NAV27) was carried out on-the-spot. Why not on-time?
C (STN00) was carried out on-time. Why not on-the-spot?

Include EventID : PLN_0_NAV
Include EventID : PLN_0_NAV27
Include EventID : STN00

(b) Only the Navigation task is displayed

Figure 6.11: Selection of Tasks To Be Displayed



C (STN00) was carried out on-time. Why not on-the-spot?
D (SEG01) was carried out on-time. Why not on-the-spot?
E (SEG01_0_NAV) was carried out perfectly.
F (SEG01_0_NAV28) was carried out on-the-spot. Why not on-time?
G (STN01) was carried out perfectly.
H (SEG02) was carried out on-time. Why not on-the-spot?
I (SEG02_0_NAV) was carried out perfectly.
J (SEG02_0_NAV29) was carried out on-the-spot. Why not on-time?
K (STN02) was carried out perfectly.
L (SEG03) was missed completely. Why?
M (SEG03_0_NAV) was carried out on-the-spot. Why not on-time?
N (SEG03_0_NAV30) was carried out on-time. Why not on-the-spot?

Include EventID : PLN_0_NAV
Include EventID : PLN_0_NAV27
Include EventID : STN00

Figure 6.12: Finding Icons For A Task

Chapter 7

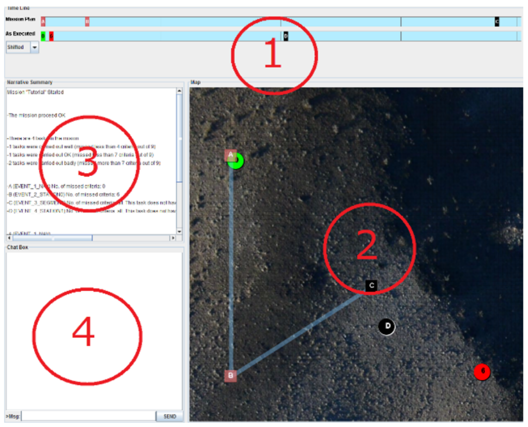
Validation

To validate the efficiency of the GNI, we conducted a user study. In this user study, we compared two user interfaces that display spatio-temporal information gathered from missions of an Unmanned Ground Vehicle (UGV) as described in Chapter 1. The first interface is the Graphical Narrative Interface (GNI) and the second interface, which we refer to as the GUI, is derived from the GNI but missing what we believed were key features in the GNI.

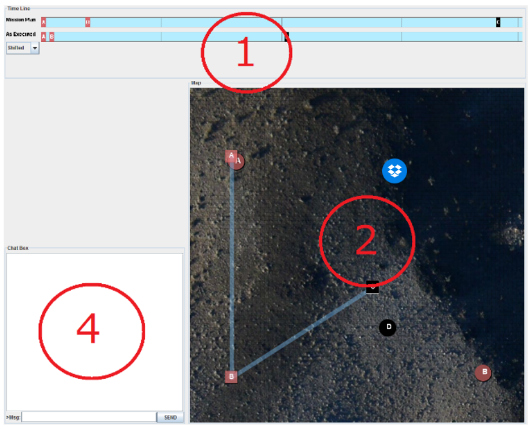
As we will show in this chapter, the GNI is more useful than the other GUI. More detailed results can be found in Section 7.3.

The GNI and GUI are illustrated in Figure 7.1. Both interfaces are equipped with a timeline (labeled as 1 on the figures), a map (label as 2), and a chat console (labeled as 4). These interfaces display information about a UGV traveling through waypoints. Both interfaces present information about both what was planned and about what was actually executed.

There are three capabilities that the GNI has but another GUI does not: 1) Displaying narrative summaries, 2) Using different colors on icons on the map and the timeline, and 3) Interlinking of the information displayed on different components of the interface. Narrative summaries are displayed in the component labeled as 3 in Figure 7.1. The text at the top of component 3 is a general and brief summary of the entire mission. As a user reads down the panel, narratives become more specific, providing more specific information about specific tasks and timings.

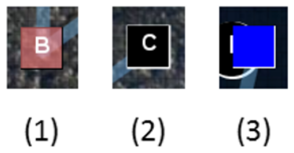


(a) GNI

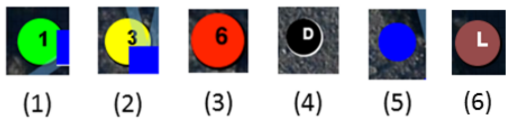


(b) GUI

Figure 7.1: Two Interfaces Used For User Study



(a) Map Icons for Planned Tasks



(b) Map Icons for Executed Tasks

Figure 7.2: List of Icons Used for User Study

The icons in Figure 7.2(a) all represent *planned tasks.* Icon 1 is the generic icon for any planned task. The embedded alphabet represents the planned order of execution. The letter "B" means the second task in a mission. Icon 2 is also a planned task, task "C" meaning the third planned task, but one that is *never executed.* Icon 3 represents a set tasks that were located very close together and that were grouped to *declutter* the display. The exact same icons are used for both the GNI and the GUI.

The icons in Figure 7.2(b) all represent *executed tasks.* The embedded numbers represent the number of performance criteria that were missed when the task was executed. The color of the icon represents a similar idea: green means executed well, yellow means OK, and red means executed poorly. So, Icon 1 missed one criterion and was executed well, Icon 2 missed three criteria and was executed OK, and Icon 3 missed 6 criteria and was executed poorly. Icon 4 represents a task that was *not planned but was added during* a mission. For this icon, the letter represents the actual order of execution. Icon 5 means some execution icons are combined to *declutter* the workspace. The GNI uses the icons 1 through 5. On the other hand, the GUI uses icons 4, 5, and 6. We already explained about icons 4 and 5. Icon 6 replaces the icons 1,2, and 3. Since the color and the number of those icons are part of the narrative summary, we do not use those for the GUI. Notice icon 6 has an alphabet letter instead of a number. The letter represents the actual order of execution.

The icons for timeline follow the same principles described above.

Interlinking means when an icon for a task is selected on a component of an interface, all the related icons for the same task will highlight on all other components

Hone et al. explained that this situation awareness can be obtained by a human user through three steps: 1) as a grasp of a little segment of the environment (Transitory Awareness or TA), 2) as an understanding of causal relationships or connections between some segments of the environment (Local Awareness or LA), and 3) as a full understanding of the entire environment (Global Awareness or GA)[22]. TA can be obtained by examining each task. LA can be obtained by studying the relationships of nearby tasks. GA can be obtained through TA and LA.

We hypothesized that the GNI provides higher SA compared to another GUI explained above, since the GNI has three additional capabilities. The narrative summaries, color and symbols for icons, and interlinking the data across different components on the interface would give better understanding about the importance and relationships between the tasks of a mission.

7.1 User Study Setup

We prepared two data sets and called them input file 1 and input file 2. We constructed these input files with the date acquired from a project that used an actual UGV. The project is called the NASA Human Exploration Telerobotics (HET) Surface Telerobotics flight demonstration[5]. Those data were provided by NASA through TRACLabs. The raw data were conditioned by removing tasks that overlapped in time with other tasks; the UGV performs only a single task at a time. Input file 2 has a larger number of tasks compared to the other, but there are no other significant differences in these input files.

Each participant uses both of these data sets on different interfaces, so there are four conditions. These conditions are described in the table below. The experiment design was a 2x2 [two interfaces (GNI and GUI) and two scenarios (file 1 and file 2)], within subjects, mixed measures design.

	Condition 1	Condition 2	Condition 3	Condition 4
First Interface to Use and Input File	GNI Input File 1	GNI Input File 2	GUI Input File 1	GUI Input File 2
Second Interface to use and Input File	GUI Input File 2	GUI Input File 1	GNI Input File 2	GNI Input File 1

Table 7.1: Starting Conditions

We measure the performance with five metrics: 1) time to complete the session 2) percentage of correct situation awareness questions (see Appendix A), 3) number of mouse clicks required to obtain necessary information to answer the questions, 4) NASA-TLX (see

Appendix D), and 5) direct comparison (see Appendix E) of two interfaces. There were eleven situation awareness questions. Four questions were designed to test the participants' level of LA. Three questions for TA, and four questions for GA. We controlled the information contained in the narrative summaries so that narratives do not provide direct answers to any of these questions. "Direct comparison" means we asked each participant to compare the two interfaces. For example, we ask which interface is easier to use or more efficient.

7.2 Procedures

There are four different parts in this user study: 1) informed consent (IRB-approved protocol) and demographic questions, 2) tutorial (see Appendix B), 3) practice (see Appendix C), and 4) actual performance measurement on both interfaces. The consent form and the protocol we followed to conduct this user study were approved by the IRB. During the tutorial, information about how to use both of the interfaces is given. Also, there are some questions to test the participant's understanding of the information provided. Participants had to get the questions correct before they were allowed to proceed to the next step of tutorial. This enforced a weak standard wherein all participants were trained to a minimum level of competence before we used their data. There was no participant who was not able to get to this level. During practice, each participant was given opportunities to answer questions in the exact same way as they were asked during the actual user study sessions. After tutorial and practice, each participant uses both the GNI and another GUI for an actual performance measurement. The whole session takes about one hour.

Note that the interfaces were referred to as Interface A and Interface B in the user study rather than as the GNI and the GUI. We did this to avoid biasing participants toward one of the interfaces.

7.3 Results and Analysis

In this section, we present the results of comparison of the GNI and GUI and analyze if those results are "significant" or not. Significance is measured by the pair-wise two-sided T-test and it is Tucky-Kramer adjusted. A comparison condition is considered significant if the p-value is less than 0.05.

Among those participant conditions, we selected the significant ones for the evaluation of the efficiency of the interfaces.

7.3.1 Demographics

Age	Min. 21 Max. 40 Ave. 27
Gender	18 males and 6 females
Profession	21 students and 3 others
English Native Speaker	12 native English speaker and 12 non-native
Frequency of using MAP tools	Multiple times a day 3, Multiple times a week 15, Multiple times a month 3, Once a month or less 3

Table 7.2: Statistics of the Participants of the User Study

Table 7.2 describe the statistics of the participants in this user study. We listed those conditions as different groups in Table 7.3 with actual numbers of participants that fit in each starting condition explained in Table 7.1.

	Group 1	Group 2	Group 3	Group 4
Combination of the order of interface to use and input file	GNI with file1 Then GUI with file2	GNI with file2 Then GUI with file1	GUI with file1 Then GNI with file2	GUI with file2 Then GNI with file1
Number of male/female participants	5/1	5/1	4/2	4/2
Number of native English speakers	3	2	3	4

Table 7.3: Distributions of Users With Different Conditions For Each Group

7.3.2 Time

Even though it is not significant, the overall average time for the GNI is less than the time for the GUI as illustrated in Table 7.3.

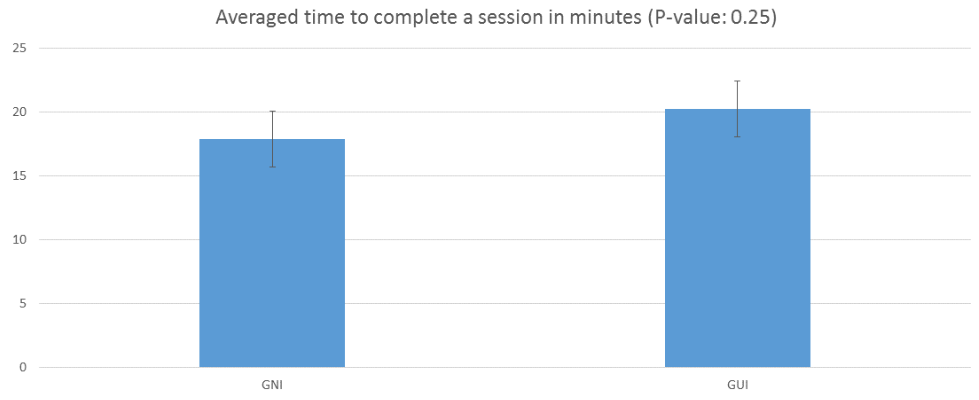


Figure 7.3: Averaged Time For Each Interface To Complete The Entire Session In Minutes

The native-English speaker participants completed the tasks significantly faster than non-native participants as illustrated in Table 7.4. However, since the number of native-English speakers for each group is about the same (see Table 7.3), so the effect of the difference in time for native and non-native participants was canceled out and did not affect to the total average time for each interface much. We think non-native English speakers had to spend more time compare to native English speakers since all the questions were given in English and also some of the questions had to be typed in English.

It seems like female participants completed the session quicker than male participants. We do not know why but there are some interesting effects related to this. Female participants made much fewer mouse clicks (see Table 7.11) compared to male participants. Besides that, the overall accuracy of the answers for female participants is slightly lower than the one for males (see Table 7.7). Some questions were skipped or some participants put answers like "I don't know". Those were considered as wrong answers. There were fourteen cases of those wrong answers. Out of those fourteen cases, eleven were made by female participants. From this result, we think that female participants tended to skip questions quicker than males

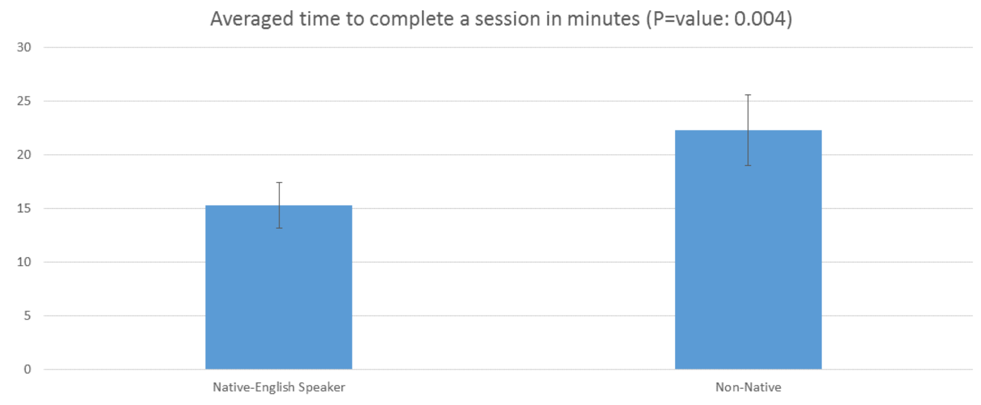


Figure 7.4: Averaged Time for Each Interface for Native Language

when they think they did not find enough information to answer the questions. Even though there is a significant difference in completion time between males and females, the effect was canceled out and did not affect the averaged completion time for each interface because of the ratio of male to female in each starting condition. There was no significant difference in completion time for different conditions illustrated in Table 7.3. The P-values for each group were between 0.55 to 0.99.

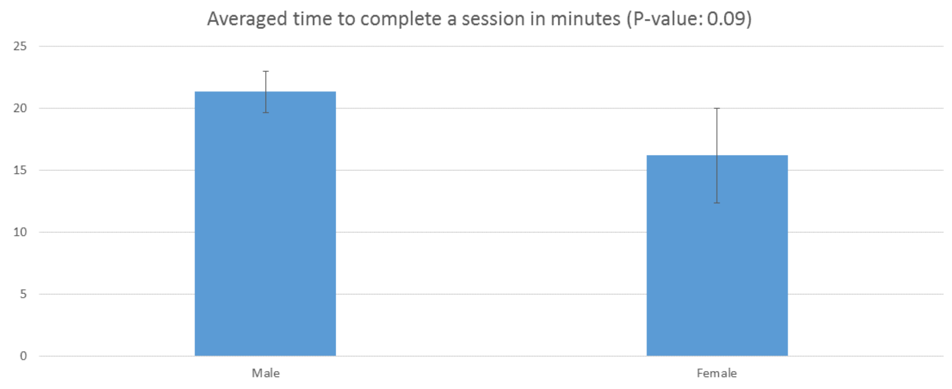


Figure 7.5: Averaged Time for Each Interface for Gender

7.3.3 Situation Awareness

There is a marginal effect on accuracy for answering questions depends on the interface type. Participants tend to answer correctly with the GNI.

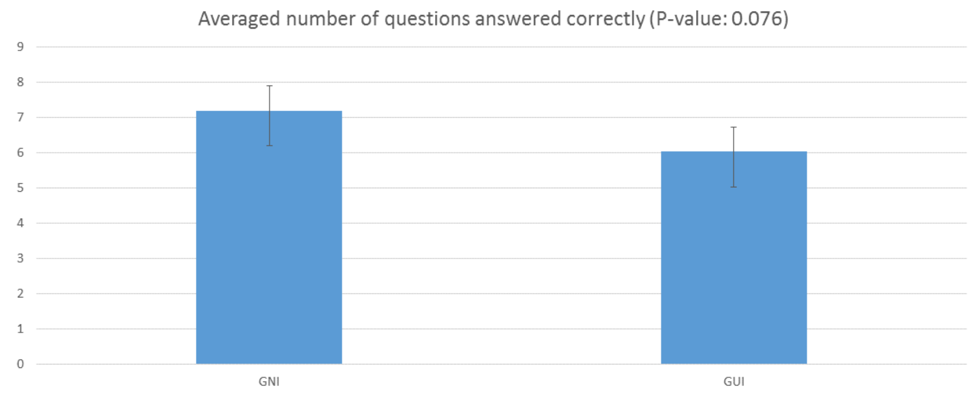


Figure 7.6: Averaged Number of Questions Answered Correctly for Interfaces

There is a slight difference in the numbers of questions participant answered correctly depends on their gender. As we stated earlier, this may have been caused by the difference in number that participants gave up to answer some questions. Female participants skipped answering questions eleven times in total where males skipped only three times in total.

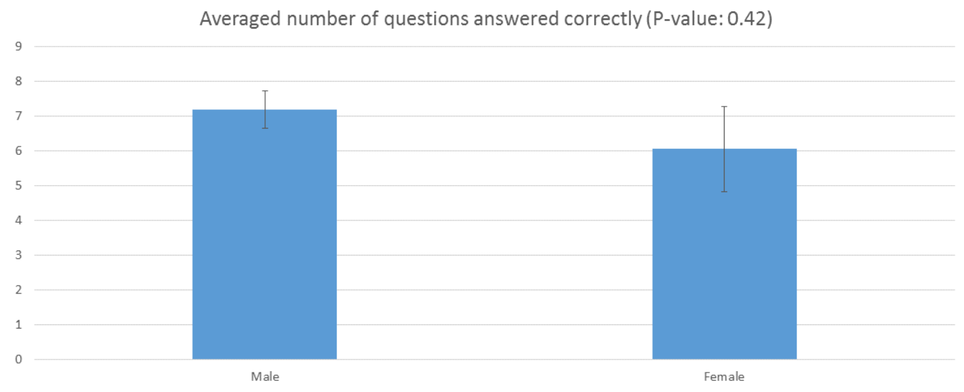


Figure 7.7: Averaged Number of Questions Answered Correctly for Gender

As illustrated in Table 7.8, participants were able to find the correct information to answer questions with the GNI more often. The GNI has higher accuracy for all the questions except No. 10.

There is a significant difference between interfaces to answer questions with a certain type of SA. As illustrated in Table 7.9, participants found out it is easier to use the GUI to find information about tasks. Notice both questions 7 and 8 are design to test the level

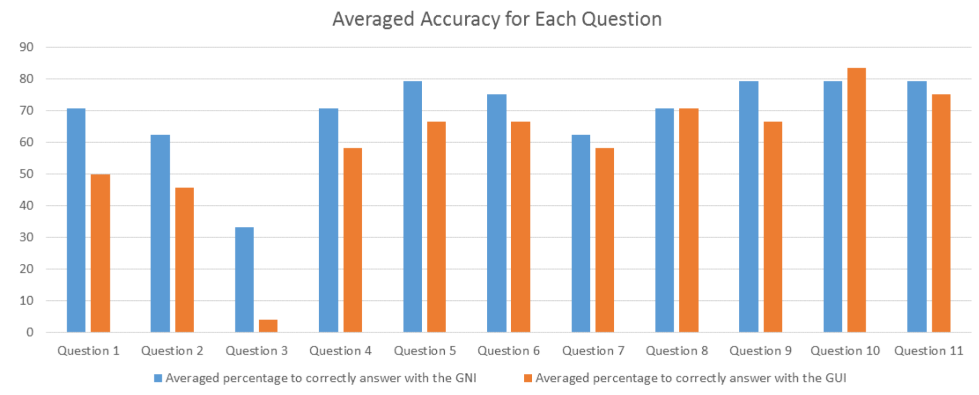


Figure 7.8: Averaged Percentage for each Question for Each Interface

of participants' LA. This may have been caused that participants tried to find the answers to those questions in the narrative summaries first. However, since they could not find the information they were looking for, they had to explore different icons to find answers. Since we do not let the direct answers to the questions appear in the narrative summaries, going through those narratives may cause some extra time to find the information necessary.

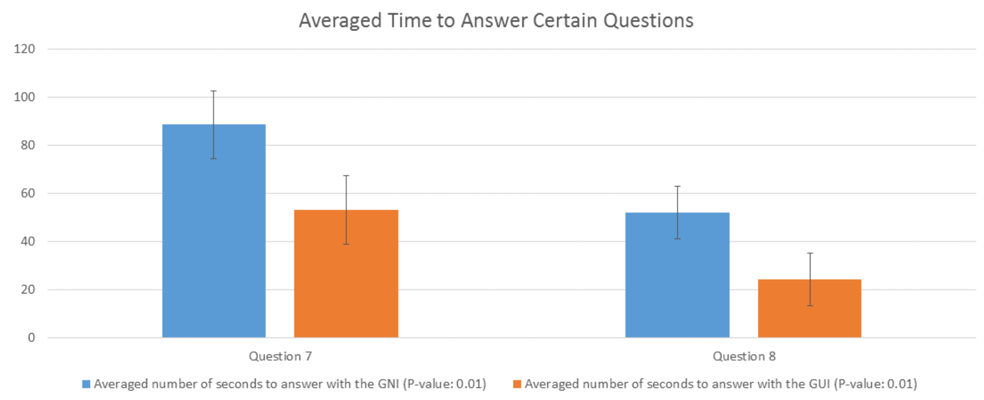


Figure 7.9: Averaged Time to Answer Questions for Different Interfaces

Question No. 5 made non-native English speakers to make more mouse clicks than native English speakers as illustrated in Table 7.10. Other than that, we did not find any significant effect for each question regardless of the type of SA.

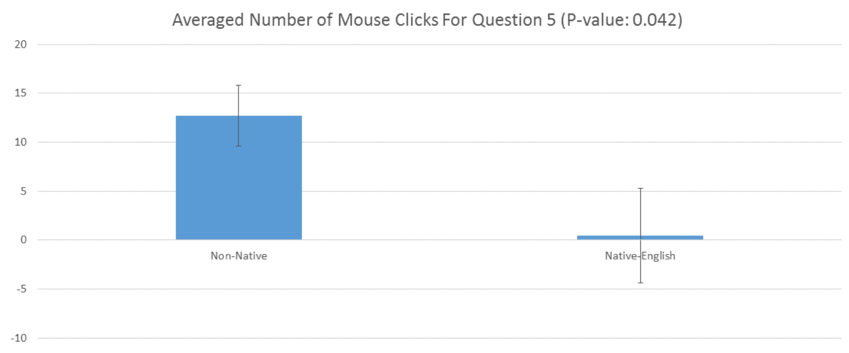


Figure 7.10: Averaged Mouse Click for Question 5 for Native Language

7.3.4 Mouse Clicks

As we stated earlier, female participants made fewer mouse clicks compared to males as illustrated in Table 7.11.

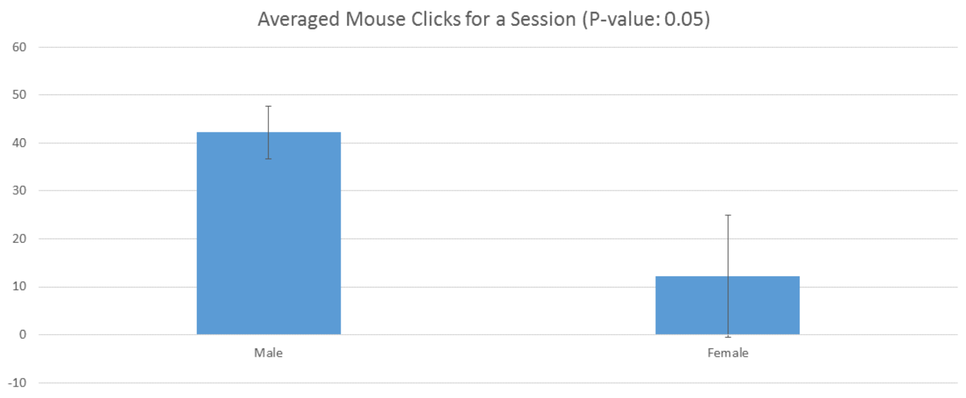


Figure 7.11: Averaged Mouse Click for a Session for Gender

7.3.5 NASA-TLX

There are six questions in NASA-TLX: 1) mental demand, 2) physical demand, 3) temporal demand, 4) performance, 5) effort, and 6) frustration. Each question is answered with a value between zero to a hundred and smaller the number is better.

Even though the GNI has better score for each question, the differences between the GNI and the GUI are not so significant as illustrated in Table 7.12.

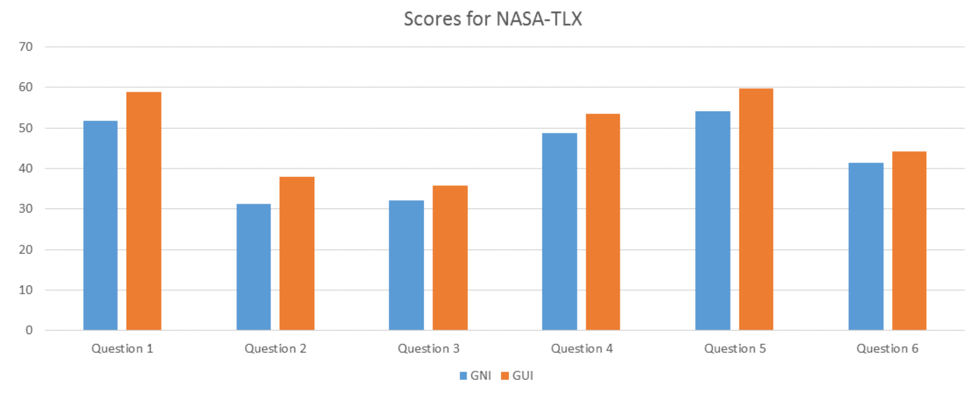


Figure 7.12: NASA-TLX

7.3.6 Direct Comparison

There are four questions in the direct comparison: 1) ease of use, 2) effectiveness, 3) preference, and 4) quality. Each question is answered with a value between zero to a hundred. Value zero means GNI.

As Table 7.13 illustrates, participants preferred the GNI over the GUI.

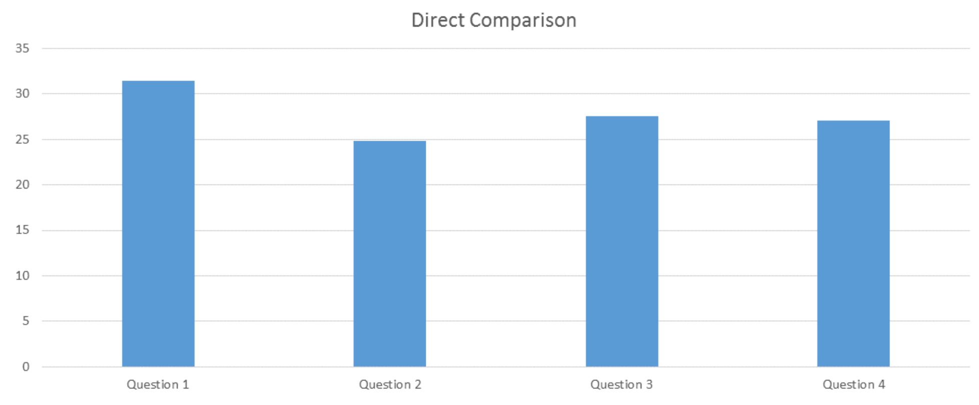


Figure 7.13: Direct Comparison

7.4 Discussion

The GNI provides information somewhat more efficiently than the GUI. Subjective workload tended to be lower for the GNI than the GUI, accuracy in answering situation awareness tended to be higher, time to complete the tasks tended to be lower for the GNI, and participants preferred the GNI over the GUI. Many of the individual results were small or only marginally significant, but together they suggest that the extra information in the narrative summaries, the extra icons, and the linkages for the GNI was useful.

However, in some cases, the GNI may require some extra time to explore since users need to go through an additional set of information, the narrative summaries.

There are some external effects of the participants in the data like gender and the native languages. The native language problem can be solved in a real situation since the language can be adjusted to the user's comfortable one. The gender effect could have been a coincidence since we used a very small number of participants.

Chapter 8

Summary and Conclusions

In this thesis, we first pointed out a problem of a human-robot team that has a highly autonomous robot working in a remote area from a human user. We then selected three key concepts (anytime summarization, storytelling, and multi-perspective analysis) to solve this problem through literature study and established a method to automatically analyze and demonstrate narrative summaries for a human use. We showed and evaluated our algorithms to acquire, store, and analyze the data in the model chapter. We explained components of the GNI and how the data would be displayed on those components in the view chapter, and then we talked about how users interact with the GNI to acquire the desired information in the controller chapter. Finally, we reviewed how the five implementation requirements are satisfied and then simulated how a user would be able to utilize the GNI in a real situation using an actual data set.

Even though much needs to be done to enhance the total quality a user interface, the prototype GNI satisfies the requirements given in Section 2. A conceptual interaction, derived from the HET project [5], illustrated how the GNI could be used to provide meaningful information to the user. We were able to find out the positive effects that the GNI provides for the users to obtain higher situation awareness compared to another user interface that does not have the tools equipped with the GNI through our user study.

Currently, the GNI is a proof of concept, meaning that future designs are likely to modify, delete, and add needed features. We listed some ideas and possible ways to increase the quality of the GNI in the following chapter as future work.

Chapter 9

Future Work

See-ability It is an important feature about displaying images to indicate the quality of the images. Information from a robot almost invariably includes data that is encoded as a function of time or as a spatio-temporal relation (i.e., what happened when and where in the world). Traditionally, this information is displayed as an x-y plot of a sequence of specific locations or geographical data. These plots implicitly encode a history and even a near-term prediction of robot activity, and thus will be included in the GNI. Morse *et al.* directly projected the quality indicator of the image onto a map and called it "see-ability" graphs [32, 33], which are similar to traditional x-y plots but include an indicator of the quality of images represented in them. We take this idea into our analysis method and display the quality information as part of the summaries.

Map We are focused on a single robot in our project, but if we control multiple robots, the use of perspective displays may help to tell stories from different perspectives as shown in Figure 9.2 [9]. Besides that, the idea of projecting additional information like the quality of images captured by a robot onto a map (as shown in Figure 9.3) would help to increase the user's situation awareness [33, 32].

Relations In section 5.2, we discussed how the information of a task that is displayed on different components of the GNI is interlinked and highlighted all together through a user's action. In our current design, all the related information of a single task can be related and highlighted. For our future work, we are planning to implement different ways to mark

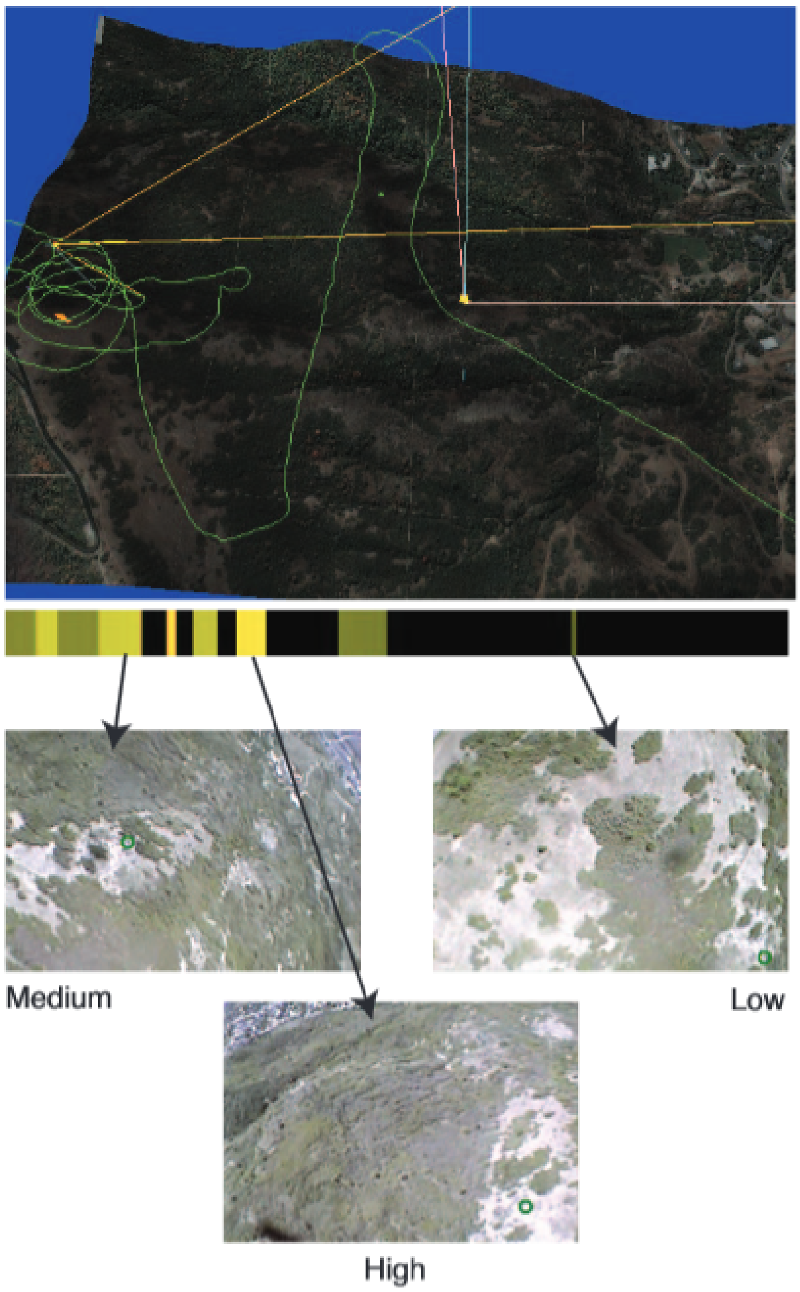


Figure 9.1: See-ability as an indexing cue. The visual indicator bar indicates which video segment includes the largest point (green marker) and the useful resolution of those segments Shading indicates quality, from bright yellow (high quality) to black (not seen). Clicking on this indicator moves directly to those frames.

and highlight icons to represent different types of relationships between multiple tasks like cause-and-effect. This means when a time delay occurred in one task, then how it affects

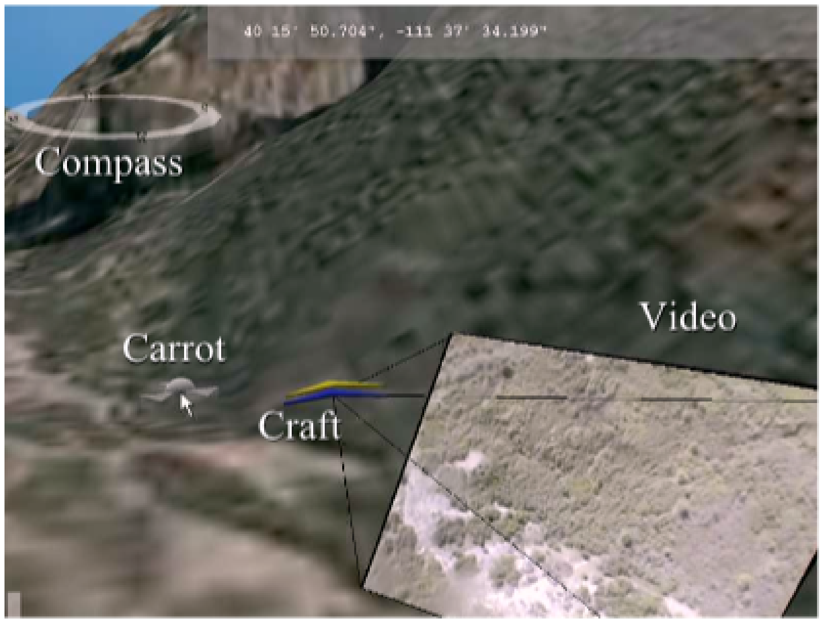


Figure 9.2: An example of a perspective display. The Unmanned-Aerial-Vehicle (UAV) is displayed in the center and heading direction and other information is also displayed accordingly.

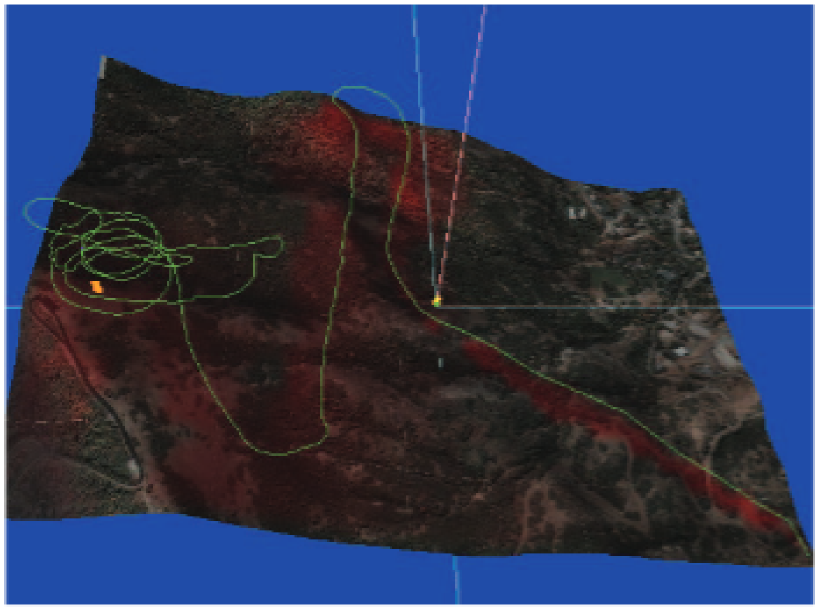


Figure 9.3: Coverage quality map overlaid in red on terrain and reference imagery. The UAV was launched from the uphill side of the terrain (orange maker on the left) and followed the flight path indicated in green.

later tasks and whether the delay gets worse or better as the mission proceeds would be useful information to display.

Applicability Currently, the GNI can only understand a very specific type of information. As we discussed in Chapter 1, we are focused on spatio-temporal information of a single UGV. Even though the input file format is very general and can be used for variety of different missions, the GNi may not understand the context of those missions and the GNI will not be able to analyze the data properly. It is because that the GNI is currently relying on the set of predefined contexts to analyze the data. For example, the GNI compares a planned and an execution time for each task. Smaller the difference between the planned and the execution is better in our missions. However, in other cases, it is possible that smaller time value (an event happens sooner) is better. To solve this issue of the GNI's limited ability to get the right context, we either include a set of contexts that are required to understand the given data as part of an input data, or we analyze a large amount of data through a technique of machine learning and build a table of universal contexts for wide variety of topics.

Appendices

Appendix A

User Study Questions

The aimed types of SA for each question are listed in square brackets.

1. Do you think this mission was successfully completed? [GA]
2. Why do you think so? [GA]
3. Tell me how many tasks were completed successfully. [LA]
4. Was the mission going better at the beginning than at the end? [TA]
5. Why do you think so/not? What do you think was the problem (at the beginning/the end)? [TA]
6. (For a particular task with a time gap) Was this time gap solely caused by this task or any other tasks caused this time gap? [TA]
7. Would you tell me how many tasks were actually carried out? [LA]
8. How many tasks were added later? [LA]
9. (Some tasks were specified.) Why do you think these tasks were added? [GA]
10. Were there any problems with the first task? [LA]
11. In this mission, many tasks were performed sooner than their planned times. Can you think of any reason? [GA]

Appendix B

Information and Questions Given to Participants During Tutorial Session

1. The purpose of this study is to compare the effectiveness of two user interfaces. You will use one interface after another to accomplish a task. While performing the task and after the task is completed, you will answer some questions to test your level of understanding about the information displayed on these interfaces.
2. Both user interfaces that you will use during this user study display information from a mission that uses an automated Unmanned Ground Vehicle (UGV). A UGV is a car-shaped robot like the one illustrated in the figure that moves around the world and gathers sensor readings at points of interest. The UGV decides where to go by following a sequence of commands generated by a human. This figure illustrates what the sequence of commands might look like if it were part of a set of directions generated by Google Maps.
3. Each user interface is equipped with a map and a time indicator as shown in the figure. Both the map and the time indicator display two types of data: "planned" and "executed". Planned data means the goals that a UVG is supposed to accomplish during a mission. Executed data means the actual data recorded by the UGV as it executes its mission, and it may differ from planned data because the execution may not go according to plan.
4. We will now explain more about each interface so that you will be able to use them. There are two different interfaces, which we call Interface A and Interface B, and you will use both of them. We'll begin your training with interface A.

5. On interface A, there are 4 components: a time indicator as indicated in No. 1, a map as indicated in No.2, a narrative summary board as indicated in No. 3, and a chat box as indicated in No. 4. We will go over each component and explain the icons used in that component.

6. This is the time indicator. The time indicator shows the order in which tasks are performed, with tasks that occur earlier shown to the left of tasks that occur later. Tasks are labeled with letters and colors, and the meaning of these letters and numbers will be explained shortly. There is no fixed scale on this time indicator, meaning that the total length of time indicated in the time indicator differs from a mission to another. The entire width of the indicator represents the total length of a mission.

7. After click on OK and close this window, click on the timeLine.

8. On the time indicator, a planned task is represented by a rectangle labeled with a letter from the alphabet. The letter "A" means that this task is supposed to be the first task executed in a mission. All planned tasks are represented by the same color rectangles, except in the following two cases: 1)when some icons are "combined" and 2) when an icon does not have a corresponding icon. We'll explain about these cases more thoroughly later. The location of the labeled rectangle on the timeline indicates when this task was supposed to begin in the mission. Some tasks, like traversing from one point to another, take time. Only the starting time is shown on the timeline, but both the time that the task starts and ends can be obtained by right-clicking the labeled rectangle. We will go over this later.

9. After click on OK and close this window, click any icon on the timeline.

10. An executed task is also represented by a rectangle, but the rectangle may be painted with a different set of colors from the set of planned tasks as well as be labeled with numbers instead of letters. Please look at the figure on the left of this dialog box and find the rectangle that is surrounded by the black circle. The green color indicates

that this task was executed successfully and the number “0” indicates that there are no problems in executing the task. To the right of this rectangle is a red rectangle labeled with the number “6”. The red color and the number “6” both indicate that this task was performed poorly. The number indicates how many demerits the robot received in accomplishing the task, so lower numbers are better. The color helps the user quickly see the relative success in performing the task. Green means good, Yellow means OK, Red means bad.

11. Notice how each rectangle on the time indicator has the same width. This is done to make it easy to see when each task starts, but it causes a problem when multiple tasks begin near the same time because it causes the rectangles to overlap. Interface A includes a tool that you can use to help manage the “clutter” that can occur when the rectangles overlap.

12. Now, we introduce a new tool on the interface. This tool is illustrated in the figure on the left of this dialog box and is found on the left side of the time indicator. The tool has a dropdown menu with three different modes to display icons on the time indicator: Shifted, Combined, and Original.

13. The default mode is called “Shifted”. In this mode, rectangles that would overlap are shifted around. More precisely, if task A follows task B but the rectangle for task A would overlap with the rectangle for task B, the task B rectangle is shifted to the right, so that the entire rectangle can be displayed. In this display mode, all tasks are guaranteed to be displayed.

14. The next mode is “Combined”. In this mode, closely placed icons are grouped into a single icon. The grouped icon is painted blue and has no letter or number label as illustrated in the figure on the left of this dialog box. Notice that since both planned and executed tasks can overlap, both timeline can include blue rectangles under the Combined mode.

15. After click on OK and close this window, select “Shifted” from the Drop Down Menu on the timeline.
16. The last mode is “Original”. In this mode, the locations of the rectangles are true to their tasks’ actual starting times, so some of the rectangles may overlap each other. By comparing these three different display modes, you may be able to obtain different perspectives about the information displayed.
17. Most of the tasks have both planned and executed data; however, it is possible for a task to miss either one of those. This means that some of the planned tasks can be skipped and some unplanned tasks can be added to a mission. In these cases, the rectangles for the tasks that are missing, either planned or executed, are colored in black.
18. In the case that a task has both planned and executed data, clicking on a rectangle icon will highlight its corresponding icon in the other timeline in orange color. In addition to this, all of the corresponding icons on other components will highlight as well.
19. Right-clicking on a rectangle will pop up a small window.
20. After click on OK and close this window, RIGHT-CLICK any icon on the timeline.
21. If you click on the entries of the popped out window, another window will pop up and display more detailed information.
22. This is the Map. Graphical representations of the mission area and locations of the waypoints or the points of interest are displayed.
23. After click on OK and close this window, click on the Map.
24. After click on OK and close this window, click any icon on the Map.
25. Now we will go over the different icons used on this component. Icons on the map are similar to what we have seen.

26. These are the planned icons. They are square and may have an alphabet letter to indicate the task ID when they are not combined. Image (1) has both planned and executed data. Image (2) has only the planned data and not executed data. Image (3) is a combined icon. When this is displayed, multiple icons are combined due to the close proximity of the planned locations fo the tasks.

27. These are the executed icons. Notice there are circles, where the planned tasks have square icons. The other important difference is the color. All the planned tasks have the same color icons unless they are grouped or having no corresponding icon. As image (4) shows, a executed task that does not have a corresponding icon has black color. All the grouped tasks have a blue icon as illustrated in image (5). All other tasks may have one of the following three colors: green, yellow, or red. These colors mean the same as explained for the icons on the time indicator.

28. Right-clicking on any of the icons will pop up a small window that displays information that is more detailed.

29. After click on OK and close this window, RIGHT-CLICK any icon on the Map

30. Left-clicking of any of the icons will higlight of the icon. If both planned and executed icons exist at the clicked location, then both icons will be highlighted.

31. Icons can be displayed in the same 3 modes explained for the time indicator. When you select one of the modes in the drop-down menu to the left of the time indicator, the layout of the icons on the map will be adjusted.

32. This is the Narrative Summary Board. It is used to display narrative summaries about the missions. This component is used only for Interface A. Narratives shown near the top are more general and brief about the entire mission. Narratives become more specific and detailed as you move towards the bottom.

33. After click on OK and close this window, click on the Narrative Summary Board.

34. Left-clicks and right-clicks on the entries of the narrative board will function the same as explained for the time indicator and the map.

35. This is the Chat Box. We do not use this for the tutorial session, but we will use it later for the actual user study session. We will display information and ask questions. You can enter your responses in the text field at the end of this component.

36. After click on OK and close this window, type in the message area of the Chat Box and click "Send".

37. That is the end of the information for Interface A. Now we move on to explain about the Interface B. Interface B is very similar to Interface A, but has fewer capabilities.

38. Now, we will explain about interface B. Interface B is very similar to inTerface A, but has less capabilities.

39. The differences between Interface A and Interface B are ... (1) interface B does not have narrative summary board, (2) icons on the time indicator and map do not have colors like green, yellow, or red. (3) icons on different components are not inter-linked. It means when you click on an icon on the map, it does not highlight the icons for the same task on different components.

Appendix C

Practice Session Questions

1. Type your name into the text box below and press SEND button
2. Read narrative summaries, left-lick on icons, use drop-down menu, right-click on an icon to display more details, and try make some other actions
3. Find the taskID for task with a letter “A”
4. Pick a task and find the planned and executed times for it to see if there is a time between the planned and executed data
5. Pick another task and find the planned and executed locations for it to see if there is a time between the planned and executed data
6. Click an icon on the map and see how the icons for the same task highlight on all the components on interface A
7. Right click on an icon and see what information displays
8. After right-click on an icon, a window pops up. Left-click on one of the entries to see more detailed information displays
9. Read the narrative summaries displayed top to bottom to see what kind of information is contained. Notice the difference in the level of details contained.
10. Try different display modes using the drop-down menu and see the difference

Appendix D

NASA-TLX

1. How mentally demanding to use this interface?
2. How physically demanding to use this interface?
3. How hurried or rushed was the pace of the task?
4. How successful were you in accomplishing what you were asked to do?
5. How hard did you have to work to accomplish your level of performance?
6. How insecure, discouraged, irritated, stressed, and annoyed were you?

Appendix E

Direct Comparison

1. Which interface was easier to use?
2. Which interface was more effective?
3. Which interface you prefer?
4. Overall, which interface was better?

References

[1] Research enrivornment for supervisory control of heterogeneous unmanned vehicles (RESCHU). `http://www.mit.edu/~yalesong/new/project.html`.

[2] Christel Baier, Joost-Pieter Katoen, et al. *Principles of model checking*, volume 26202649. MIT press Cambridge, 2008.

[3] Armin Biere, Alessandro Cimatti, Edmund Clarke, and Yunshan Zhu. *Symbolic model checking without BDDs*. Springer, 1999.

[4] James Bruce and Manuela Veloso. Real-time randomized path planning for robot navigation. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2383–2388. IEEE, 2002.

[5] Maria Bualat, Debra Schreckenghost, Estrellina Pacis, Terrence Fong, Donald Kalar, and Brent Beutter. Results from testing crew-controlled surface telerobotics on the international space station.

[6] Jean M Catanzaro, Matthew R Risser, John W Gwynne, and Daniel I Manes. Military situation awareness: Facilitating critical event detection in chat. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 50, pages 560–564. SAGE Publications, 2006.

[7] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 8(2):244–263, 1986.

[8] Malcolm E Cook, Wendy G Lehnert, and David D McDonald. Conveying implicit content in narrative summaries. In *Proceedings of the 10th international conference on Computational linguistics*, pages 5–7. Association for Computational Linguistics, 1984.

[9] Joseph Cooper and Michael A Goodrich. Towards combining uav and sensor operator roles in uav-enabled visual search. In *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, pages 351–358. IEEE, 2008.

[10] Jacob W Crandall and ML Cummings. Developing performance metrics for the supervisory control of multiple robots. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 33–40. ACM, 2007.

[11] Jacob W Crandall, Michael A Goodrich, Dan R Olsen Jr, and Curtis W Nielsen. Validating human-robot interaction schemes in multitasking environments. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(4):438–449, 2005.

[12] Trevor Davies and Amor Jnifene. Multiple waypoint path planning for a mobile robot using genetic algorithms. In *Computational Intelligence for Measurement Systems and Applications, Proceedings of 2006 IEEE International Conference on*, pages 21–26. IEEE, 2006.

[13] Seniz Demir, Sandra Carberry, and Kathleen F McCoy. Generating textual summaries of bar charts. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 7–15. Association for Computational Linguistics, 2008.

[14] Peter F Drucker. *The practice of management.* Allied Publishers, 1975.

[15] Mica Endsley. Theoretical underpinnings of situation awareness: A critical review. *Situation awareness analysis and measurement*, pages 3–32, 2000.

[16] Mica R Endsley. Situation awareness global assessment technique (sagat). In *Aerospace and Electronics Conference, 1988. NAECON 1988., Proceedings of the IEEE 1988 National*, pages 789–795. IEEE, 1988.

[17] Stephen M Fiore, Rudy McDaniel, and Florian Jentsch. Narrative-based collaboration systems for distributed teams: Nine research questions for information managers. *Information Systems Management*, 26(1):28–38, 2009.

[18] Jeremiah Gertler. Us unmanned aerial systems. DTIC Document, 2012.

[19] Michael A Goodrich and Alan C Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

[20] Justin G Hollands and Christopher D Wickens. *Engineering psychology and human performance.* Prentice Hall New Jersey, 1999.

[21] Bill R Hollifield and Eddie Habibi. *Alarm management: Seven effective methods for optimum performance.* ISA, 2007.

[22] Geoffrey Hone, Lynne Martin, and Robert Ayres. Awareness–does the acronym âĂIJSAâĂİ still have any value. In *Proceedings of the 11th International Command and Control Research and Technology Symposium: Coalition Command and Control in the Networked Era. held at Cambridge UK. Washington*, 2006.

[23] Joan H Johnston, Stephen M Fiore, and Rudy McDaniel. Applying the narrative form and xml metadata to debriefing simulation-based exercises. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 49, pages 2135–2139. SAGE Publications, 2005.

[24] Peter GW Keen. Decision support systems: the next decade. *Decision Support Systems*, 3(3):253–265, 1987.

[25] Jesper Kjeldskov, Mikael B Skov, and Jan Stage. Instant data analysis: Conducting usability evaluations in a day. In *Proceedings of the third Nordic conference on Human-computer interaction*, pages 233–240. ACM, 2004.

[26] Gary Klein, Rebecca M Pliske, Beth Crandall, and David Woods. Features of problem detection. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 43, pages 133–137. SAGE Publications, 1999.

[27] Glenn E Krasner, Stephen T Pope, et al. A description of the model-view-controller user interface paradigm in the Smalltalk-80 system. *Journal of object oriented programming*, 1(3):26–49, 1988.

[28] Ying Li, Shih-Hung Lee, Chia-Hung Yeh, and C-CJ Kuo. Techniques for movie content analysis and skimming: Tutorial and overview on video abstraction techniques. *Signal Processing Magazine, IEEE*, 23(2):79–89, 2006.

[29] David M Mark. Human spatial cognition. *Human factors in geographical information systems*, pages 51–60, 1993.

[30] Teenie Matlock. Fictive motion as cognitive simulation. *Memory & Cognition*, 32(8):1389–1400, 2004.

[31] James McLurkin, Jennifer Smith, James Frankel, David Sotkowitz, David Blau, and Brian Schmidt. Speaking swarmish: Human-robot interface design for large swarms of autonomous mobile robots. In *AAAI Spring Symposium: To Boldly Go Where No Human-Robot Team Has Gone Before*, pages 72–75, 2006.

[32] Bryan S. Morse, Cameron H. Engh, and Michael A. Goodrich. UAV video coverage quality maps and prioritized indexing for wilderness search and rescue. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, HRI '10, pages 227–234, Piscataway, NJ, USA, 2010. IEEE Press.

[33] Bryan S Morse, Damon Gerhardt, Cameron Engh, Michael A Goodrich, Nathan Rasmussen, Daniel Thornton, and Dennis Eggett. Application and evaluation of spatiotemporal enhancement of live aerial video using temporally local mosaics. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[34] Tsuyoshi Muramoto. A proposal of a supporting method to implement an iterative multi-perspective story. Master's thesis, Tokyo University of Technology, Tokyo, Japan, 2008.

[35] Curtis W Nielsen. *Using augmented virtuality to improve human-robot interactions*. PhD thesis, Citeseer, 2006.

[36] C.W. Nielsen, M.A. Goodrich, and R.W. Ricks. Ecological interfaces for improving mobile robot teleoperation. *Robotics, IEEE Transactions on*, 23(5):927–941, 2007.

[37] Dan R. Olsen, Jr. and Stephen Bart Wood. Fan-out: Measuring human control of multiple robots. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 231–238, New York, NY, USA, 2004. ACM.

[38] R. Parasuraman, T.B. Sheridan, and Christopher D. Wickens. A model for types and levels of human interaction with automation. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 30(3):286–297, May 2000.

[39] Heekyong Park and Jinwook Choi. V-model: a new innovative model to chronologically visualize narrative clinical texts. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, pages 453–462. ACM, 2012.

[40] Beverly Plester, Jenna Richards, Mark Blades, and Christopher Spencer. Young children's ability to use aerial photographs as maps. *Journal of Environmental Psychology*, 22(1):29–47, 2002.

[41] Ehud Reiter. Nlg vs. templates. *arXiv preprint cmp-lg/9504013*, 1995.

[42] Theresa Marie Rhyne, Alan MacEachren, and Theresa-Marie Rhyne. Visualizing geospatial data. In *ACM SIGGRAPH 2004 Course Notes*, page 31. ACM, 2004.

[43] Jean Scholtz and S Bahrami. Human-robot interaction: Development of an evaluation methodology for the bystander role of interaction. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, volume 4, pages 3212–3217. IEEE, 2003.

[44] D. Schreckenghost, T. Fong, T. Milam, E. Pacis, and H. Utz. Real-time assessment of robot performance during remote exploration operations. In *Aerospace conference, 2009 IEEE*, pages 1–13, 2009.

[45] Debra Schreckenghost, Terrence Fong, Hans Utz, and Tod Milam. Measuring robot performance in real-time for nasa robotic reconnaissance operations. In *Proceedings of the 9th Workshop on Performance Metrics for Intelligent Systems*, PerMIS '09, pages 194–202, New York, NY, USA, 2009. ACM.

[46] Debra Schreckenghost, Tod Milam, and Terrence Fong. A perspective on human-robot interaction for nasa's human exploration missions. In *2014 AAAI Fall Symposium Series*, 2014.

[47] Thomas B Sheridan. *Telerobotics, automation, and human supervisory control.* MIT press, One Rogers Street Campbridge MA, 1992.

[48] Ben Shneiderman. A taxonomy and rule base for the selection of interaction styles. *Human factors for informatics usability*, page 325, 1991.

[49] Daniel J Simons and Ronald A Rensink. Change blindness: Past, present, and future. *Trends in cognitive sciences*, 9(1):16–20, 2005.

[50] Harvey S Smallman and Mark John. Naive realism: Limits of realism as a display principle. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 49, pages 1564–1568. SAGE Publications, 2005.

[51] Harvey S Smallman and Mark St John. Chex (change history explicit): New HCI concepts for change awareness. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 47, pages 528–532. SAGE Publications, 2003.

[52] Matthew G Styles, Shin-Young Jung, Chihiro Eto, and Geoffrey M Draper. An approach to hypertext fiction for mobile devices. In *Proceedings of the 2nd workshop on Narrative and hypertext*, pages 1–6. ACM, 2012.

[53] S Shyam Sundar, Qian Xu, and Saraswathi Bellur. Designing interactivity in media interfaces: A communications perspective. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 2247–2256. ACM, 2010.

[54] Mariët Theune, Nanda Slabbers, and Feikje Hielkema. The narrator: NLG for digital storytelling. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 109–112. Association for Computational Linguistics, 2007.

[55] Kavita E Thomas and Somayajulu Sripada. Atlas.txt: Linking geo-referenced data to text for NLG. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 163–166. Association for Computational Linguistics, 2007.

[56] Charles Wiecha, William Bennett, Stephen Boies, John Gould, and Sharon Greene. ITS: A tool for rapidly developing interactive applications. *ACM Trans. Inf. Syst.*, 8(3):204–236, July 1990.

[57] Michael F. Worboys. A unified model for spatial and temporal information. *The Computer Journal*, 37(1):26–34, 1994.

[58] Sarah Worth. Narrative knowledge: knowing through storytelling. *Furman University, Greenville, South Carolina*, 2006.

[59] Sarah E Worth. Storytelling and narrative knowing: An examination of the epistemic benefits of well-told stories. *The Journal of Aesthetic Education*, 42(3):42–56, 2008.