Designing Interactive Experiences to Explore Artwork Collections: a Multimedia Dialogue System Supporting Visits in Museum Exhibits

Antonio Origlia[1,2], Enrico Leone[1], Antonio Sorgente[2], Paolo Vanacore[2], Maria Parascandolo[1], Francesco Mele[2], and Francesco Cutugno[1,2]

[1] PRISCA-Lab, Federico II University, Naples, Italy,
{antonio.origlia, enrico.leone, maria.parascandolo, cutugno}@unina.it,
[2] Institute of Applied Sciences and Intelligent Systems - CNR, Naples, Italy,
{a.sorgente, p.vanacore, f.mele}@isasi.cnr.it

Abstract. Speech and natural language processing have a central role in the implementation of systems designed to make the museum more reactive to users' inputs and to improve the overall interaction quality. In this paper, we present the design and implementation of a dialogue system to provide multimedia presentations for museum visits. A corpus of speech recordings in Italian was collected with a mobile application to obtain a reference set of possible ways for the users to express their intentions. On the basis of this corpus, a set of recurring syntactic patterns associated to device requests was extracted to let the dialogue system separate device commands from information queries. Disambiguation strategies depending on the context are also applied in presence of partial syntactic patterns. Information queries are answered by automatically assembling portions of semantically annotated texts and are synchronized with relevant multimedia resources. A case study on the '800 exhibit at the Capodimonte museum in Naples is presented[3].

Keywords: dialogue systems, cultural heritage

1 Introduction

Italian has a very poor visibility in the area of spoken dialogue systems basic research. The EVALITA evaluation campaign held in 2009 [1] showed the state of art for telephonic systems was limited to the features offered by VoiceXML standard. Participants in that evaluation campaign were able to set up three different system initiative dialogue managers in the field of train services (ticketing, booking and timetable queries) but performances were found to be below the performance that is usually obtained on English. Also, recent dialogue systems for Italian equipped with semantic reasoning capabilities were presented in [2–4], but they only considers chat based interaction. In the passage from telephonic

[3] This work is supported by the Italian PAC project *Cultural Heritage Emotional Experience See-Through Eyewear* (CHEESE).

to mobile applications first and to generalized spoken language understanding systems, most of the Italian researchers participated into international projects and mainly worked on languages different from their own. This is the case, for example, of the recently published SPEAKY [5] development environment for robotic vocal interfaces. At the same time big companies that were investing in personal assistant mobile apps and similar products, extended their native solutions to our language following industrial procedures that did not give raise to knowledge to be spread in the scientific community. In this paper, we will describe the development of a dialogue system integrated with remote augmented reality interfaces in a cultural heritage setting. We will include a brief description of the problems that arise when dealing with delicate environments, like the '800 exhibit in the Capodimonte museum. These pose serious limitations to technological interventions that have an impact on the overall design process. We will describe how these were addressed and the how the system architecture was implemented.

2 Material

An important problem that arises when working with an environment that requires technology to be non-invasive, like a museum exhibit, is that it is difficult to involve end users in the early steps of system development. Exhibits may not be always open to the general public, estimated visitor attendance can vary due to external conditions and wifi connectivity is not always guaranteed. In our case study, the '800 exhibit at the Capodimonte museum in Naples, wifi connection presents a problem of its own as the exhibit is located inside the Bourbon Royal Palace, where walls are very thick and the possibility of intervention are limited. For this reason, in order to obtain a reference set of possible ways for the users to access the system's functions before this is deployed, we used a simple prototype application to collect speech utterances and design the dialogue system accordingly.

The application is implemented as an Android app running on a smartphone in uncontrolled environments. To avoid influencing participants in producing always the same utterances and obtain a higher expressive variability, we chose to present the scenarios using an iconographic approach. Each participant, at each step, was prompted with a set of icons representing a specific user request. An example of the *VolumeUp* scenario is presented in Figure 1. The participant can record the utterance, listen to it and submit it to the remote collection server when she is satisfied. Scenarios cover both device commands (volume control, taking pictures, recording videos...) and content-related queries. If a prompt was not clear, the user would tap the single icons to visualize a single word explaining the icon. This way, no suggestion about how to combine the icons to derive the scenario were given. The users were, in general, able to derive the meaning of the prompt. After a manual check, only 171 utterances have been discarded because the users provided inconsistent recordings.

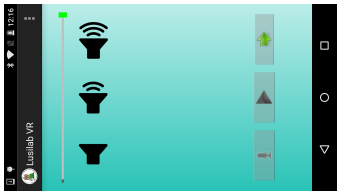

Fig. 1. A screenshot of the mobile app used to collect the reference corpus.

The prompts were randomly presented to the users and were proposed five times each to encourage people to provide multiple ways to ask for the same service, increasing expressive variability. 22 gender-balanced participants with good technological competence were recruited and 17 scenarios were foreseen. A total number of 1870 recordings were collected this way. An expert linguist listened to the material and provided the correct transcription to obtain an estimate of the ASR errors. After a manual check, the correct transcription was found as 1-best in 70% of the cases. In 11% of the correct transcription was either presented a 2-best or 3-best. In the remaining 19% of the cases, the Google ASR engine was not able to provide the correct transcription on the first 3-best. This is mainly caused by the variable quality of the recordings and it represents a good approximation of the performance we can expect from the ASR. Also, it indicate that a good number of cases may be recovered by applying re-ranking techniques. A common error committed by the ASR engine was providing the transcription "Firma l'audioguida" (Sign the audioguide) as the 1-best while the correct "Ferma l'audioguida" (Stop the audioguide) was found as 2-best. In our context, of course, "Ferma l'audioguida" makes more sense than "Firma l'audioguida" and would be recovered, ideally bringing the system's performance around 81% correct transcription.

Concerning the question answering system, museum experts provided a collection of texts and media related the '800 exhibit. This material contains textual information describing 4 museum rooms and 7 artworks and it also contains 123 media objects linked to the relevant parts of the reference texts. This allows the question answering system to control the timing of potential accompanying media presentations when the answer to a question is assembled.

3 System architecture

In this section, we describe the client-server architecture used to deploy the CHEESE system. Most of the logic is located on the server side but some issues related to the client and how these were managed are worth mentioning.

Client side. Although the dialogue management is independent from the client interface, limits due to the chosen wearable device influence the configuration of the speech manager. In our case, we use the Epson Moverio BT-200 glasses for augmented reality, which are equipped with Android 4.0.4. This is an important

issue as, in this version of Android, no offline recognition support is offered by the system. For this reason, we developed an Android app that continuously listens to the microphone and streams audio towards the server. On server side, an audio acquisition thread collects the recorded input and applies Voice Activity Detection before connecting to Google Speech to obtain the transcription. To perform audio streaming and segmentation, adintool, which is part of the Julius ASR engine [6], is used. In order to connect the two parts of the system, the audio streaming procedure replicates, in Java, the C++ procedure used by adintool when operating in client mode. As future versions of wearable glasses will be equipped with higher versions of Android supporting offline recognition, the system can also be configured to manage strings instead of audio.

Server side. The server side of the dialogue system is centered on the Opendial framework [7], which provides a flexible environment to design dialogue systems using and XML-based language and can also be extended with customized plugins using Java. Opendial represents the dialogue state as a set of variables and it lets the user define a series of internal models triggered by variable updates that automatically produce reactions accordingly to the the observed state.

Although not mandatory, there are typically three main models in an Opendial application. The *Natural Language Understanding* (NLU) model analyzes the user input and it maps it on a finite set of possible user actions. The *Action Selection Model* (ASM) connects the user action to the correspondent machine action. The *Natural Language Generation* (NLG) model produces spoken content in accordance with the selected machine action. In the CHEESE framework, we have three separate NLU models to handle different moments of the interaction: the first separates commands to the device (volume control, taking pictures or videos, etc.) from user queries concerning cultural heritage items that are part of the considered exhibit. The second detects requests for device-related functions. The third one detects incomplete commands and summarizes the possible outcomes so that clarification strategies can be applied to recover the interaction.

As the system is focused on Italian, a set of plugins to include, in Opendial, the following software tools, needed to process this language: a plugin to receive the audio stream from the client and transcribe it using Google Speech; a plugin to obtain POS tags from the Treetagger[8] tool; a plugin to normalize the utterance substituting synonyms of target terms with the target term itself; a plugin to extract the dependency-based parse tree of the normalised utterance using the Turin University Linguistic Environment (TULE)[9]; a plugin to connect Opendial to the higher-level system handling user queries.

Concerning the last plugin, a communication protocol based on JavaScript Object Notation (JSON) has been adopted. The JSON string contains the multimedia response for the user and defines the synchronisation of synthesised text and media. Its structure is based on a simplified version of Synchronized Multimedia Integration Language (SMIL)[4]. The main modules used for the in-

[4] https://www.w3.org/TR/REC-smil/

terpretation of user requests are a parser to identify its grammatical structure, a set of semantic services implemented for the detection of semantic concepts in the text such as events, entities, locations, etc, and services to access semantic repository as MultiWordnet[10] and Wiktionary (https://www.wiktionary.org/).

4 Dialogue System

In this Section, we describe the dialogue management logic governing the system. The system is modular and it applies a pipeline process after receiving an input utterance to evaluate its content and plan a reaction. First of all, the input string is preprocessed to obtain a normalised utterance using the plugins described in the previous Section. The dialogue state is then updated considering the incoming utterance and the current position of the user, which is provided externally and is relevant to answer queries like "Chi l'autore di questo quadro" (Who is the author of this painting). The NLU model dedicated to the detection of WH-questions is the first to run. If this model detects a WH-question, the ASM gives control to the question answering system, otherwise the NLU command detection model is run. If a syntactic pattern associated with a device command is detected, the ASM activates the corresponding device action, otherwise the NLU model for recover strategies is activated. This model checks if incomplete syntactic patterns can be detected in the utterance and, if this is the case, the ASM instructs the NLG model to pose an appropriate question to the user to disambiguate the command. This module also attempts to resolve ambiguities based on the current context. If no partial syntactic pattern can be found, the system asks the user to confirm the automatic transcription and, if this is confirmed, aborts the interaction as it is not able to help.

4.1 Command/query separation

In our approach, we have focused on wh-question (or content questions), queries containing a question word (called wh-words) such as, for example, 'chi' (who) or 'quando' (when) [11]. For the identification of wh-questions, a set of lexico-syntactic patterns are defined. These are implemented as regular expressions detecting direct and indirect queries and identifying specific linguistic expressions. From the analysis of the reference corpus we observed that many users do not just make specific requests, but also use general queries such as "*dammi altre informazioni*" (give me more information), "*mi dici qualcosa sul quadro*" (can you tell me something about the picture).

4.2 Device commands management

Starting from the material collected with the procedure described in Section 2, an expert linguist analysed the output of the Treetagger and TULE tools applied to the normalised utterances to obtain the Opendial model dedicated to command recognition. Rules in this model attempt to match recurring syntactic patterns

related to specific user requests against the dependency parsing tree obtained from the received utterance. The rules consider the presence of a variable length syntactic dependence, in the tree, between a set of target subtree roots (usually verbs), covering the ones observed in the corpus, and, possibly, a set of target terms (usually nouns) in the tree. For both target roots and terms, it is possible to admit their MWN synsets. It is possible to summarize the structure of these rules as a tuple $< R, T, l, S >$, where R is a set of target subtree roots, T is a set of optional target terms, l is the maximum length of the dependency chain linking a member of R to a member of T and S is a subset of the union of R and T containing the terms for which synonyms are accepted. If T is empty, l is always 0 and the terms included in R must be the root of the entire tree for the rule to be matched. Multiple tuples can be associated with a single command to describe different syntactic patterns. For example, the *TakePicture* command is associated with the tuples

$$< \{scattare, \ldots, foto\}, \{\}, 0, \{scattare, \ldots, foto\} >$$
$$< \{fare\}, \{foto\}, 2, \{foto\} >$$

The first tuple handles the cases in which users use isolated words, like imperative verb forms, to control the device ("scatta!", "fotografa!", "foto!"), the second tuple handles the case of the most natural sentence "fai una fotografia" (Take a picture) where *fotografia* is a synonym of *foto*(photo) and also covers the frequent use of *Googlese*(talk using keywords) by the users as, in the corpus, utterances like "fai foto" were not uncommon. This is also the reason why dependency types are not included in the rules: we observed that, when users shift to *Googlese*, the structure of the dependency tree is preserved in most cases but the reported relationship types are erratic.

When no command pattern can be matched exactly, the system checks for partial matches in the preceding rules represented by the presence of target terms in the input utterance to recover the interaction. In this phase, the system also checks the active processes to reduce the set of possibilities when binary actions are considered. For example, if the target term "registrazione" is detected and the device is already recording a video, the only possible action related to the target term is *RecStop*, which stops the recording. The system asks for confirmation before performing the action as a safety measure. When context does not help to reduce the set of possible actions to one, as in the case of "Avvia!"(Start) when both video recording and the audioguide are not running, the system prompts the user to specify an action among the possible ones to proceed. When system prompts conclude an interaction turn, control is given directly to the system query handler to complete the interaction after processing the user utterance answering the question.

4.3 Question answering

In this section we introduce the approach used to understand the user's request and generate the response that satisfies it. First, we check if the sentence is a Wh-query type. Then, if the check has success, a set of rules are applied to detect the

topic and the information request about topic. Discovered the type of information request, queries are generated to research answers in a knowledge base. This latter is composed by stories that have been annotated through an event-based formalism. Finally, texts and media retrieved are composed to generate an unique multimedia response to be returned to the user.

For the extraction of the topic and the arguments of the request, we have defined a set of rules based on the relations contained in the dependency tree of the sentence and contextual information of user's position. The rules defined are used for discovering the topic (subject which is discussed) and about what the visitor asks. The recognition process identifies the components of the query using the same semantics adopted to annotate the story of an artwork. For this, analysing the dependency tree and the topic of the query, we discover the events from their components: the action (what), the location (where), the participants (who), and the interval of happening (when).

For example: "*Chi ha dipinto la Morte di Cesare*(Who painted Cesar's Death)" is related to the action "dipingere (paint)" and participant "la morte di Cesare". Also, the visitor want know the author ("Chi (Who)"). In this example, the rules used for detected the action and the participant are:

$$verb(S, V) \wedge \sim auxiliary(S, V) \rightarrow action(S, V)$$

$$verb(S, V) \wedge obj(S, V, O) \rightarrow participant(S, O)$$

where $verb(S, V)$ means that V is the verb of the sentence S, $auxiliary(S, V)$ means V is an auxiliary verb of S and $obj(S, V, O)$ means O is the object if V in S. In this example, the topic is explicit and corresponds to the object of verb. If the query has a passive form ("*da chi stata dipinta la Morte di Cesare*") the participant is the subject. There are other rules to discovery the location and/or time interval of the event. If the topic is not explicit, is detected from contextual information analysing the user's position.

To assemble the multimedia response, we first have to obtain a textual answer. To do this, we adopt a modified version of the system presented in [12]. All the histories and media relative to an exhibit are archived in a semantic repository annotated with an event-based formalism. In this work, this formalism is the Cultural Story Web Language (CSWL)[13], which represents. Starting from the results of query interpretation, we reformulate the user request as a CSWL query, and apply query expansion using semantic lexical databases (MultiWordnet and Wiktionary). The list of answers (events) obtained from query results are ranked and the best answer is expanded through extra events correlated with it, if necessary. Then, the corresponding text associated to the selected events, and the relative media recovered in the repository, are assembled as a presentation. The process takes in account the semantic annotation associated to the elements (texts and media) and synchronizes them so that media items are visualised coherently with the relevant time instants in which a synthetic voice is talking about the content it represents[14].

5 Conclusions

We presented the design and implementation of a spoken dialogue system for Italian aiming to assist the visitors of a museum exhibit. We reported the difficulties encountered in designing an interactive system using natural language understanding caused by the particular environment of museums located in historical places and how we addressed them. We also described the material collected to obtain a first working prototype of the CHEESE system. The system architecture is flexible, modular and can easily be adapted to future updates due to upcoming technologies. Also, spoken dialogue systems for Italian are not common and our contribution explores an area of recent interest for interactive environments, cultural heritage, which may find relevant applications in Italy. Future work will consist in collecting on-site feedback to evaluate the system and take full advantage of the probabilistic environment offered by Opendial to fine-tune the system.

References

1. Baggia, P., Cutugno, F., Danieli, M., Pieraccini, R., Quarteroni, S., Riccardi, G., Roberti, P.: The multi-site 2009 evalita spoken dialog system evaluation. In: Proc. of EVALITA. (2009)
2. Sorgente, A., Brancati, N., Giannone, C., Zanzotto, F.M., Mele, F., Basili, R.: Chatting to personalize and plan cultural itineraries. In: UMAP Workshops. (2013)
3. Stock, O.: Language-based interfaces and their application for cultural tourism. AI Magazine **22**(1) (2001) 85
4. Stock, O., Zancanaro, M.: PEACH-Intelligent interfaces for museum visits. Springer Science & Business Media (2007)
5. Bastianelli, E., Nardi, D., Aiello, L.C., Giacomelli, F., Manes, N.: Speaky for robots: the development of vocal interfaces for robotic applications. Applied Intelligence **44**(1) (2015) 43–66
6. Lee, A., Kawahara, T.: Recent development of open-source speech recognition engine julius. In: Proc. of APSIPA ASC. (2009) 131–137
7. Lison, P.: A hybrid approach to dialogue management based on probabilistic rules. Computer Speech & Language **34**(1) (2015) 232 – 255
8. Schmid, H.: Improvements in part-of-speech tagging with an application to german. In: Proc. of the ACL SIGDAT-Workshop. (1995) 47–50
9. Lesmo, L.: The rule-based parser of the nlp group of the university of torino. Intelligenza artificiale **2**(4) (2007) 46–47
10. Pianta, E., Bentivogli, L., Girardi, C.: Developing an aligned multilingual database. In: Proc. 1st Intl Conference on Global WordNet. (2002)
11. Rossano, F.: Questioning and responding in italian. Journal of Pragmatics **42**(10) (2010) 2756 – 2771
12. Mele, F., Sorgente, A.: Semantic mashups of multimedia cultural stories. Intelligenza Artificiale **6**(1) (2012) 19–40
13. Mele, F., Sorgente, A.: CSWL - un formalismo per rappresentare storie culturali nel web. Technical Report 180/15, Inst. of Cybernetics "E. Caianiello", CNR (2015)
14. Sorgente, A., Vanacore, P., Origlia, A., Leone, E., Cutugno, F., Mele, F.: Multimedia responses in natural language dialogues. In: Proceedings of AVI*CH 2016. Volume 1621 of CEUR Workshop Proceedings., CEUR-WS.org (2016) 15–18