Coordinating Speech with Touch Input and Visual Cues in Human–Robot Interaction: A Multimodal System Evaluated through Metamorphic Testing

Massimo Donini
massimo.donini@unito.it
Computer Science Dept.
University of Turin, Italy

Paolo Arcaini
arcaini@nii.ac.jp
National Institute of Informatics
Tokyo, Japan

Michael Oliverio
michael.oliverio@unito.it
Computer Science Dept.
University of Turin, Italy

Fuyuki Ishikawa
f-ishikawa@nii.ac.jp
National Institute of Informatics
Tokyo, Japan

Alessandro Mazzei
alessandro.mazzei@unito.it
Computer Science Dept.
University of Turin, Italy

Deyun Lyu
lyudeyun@nii.ac.jp
National Institute of Informatics
Tokyo, Japan

Cristina Gena
cristina.gena@unito.it
Computer Science Dept.
University of Turin, Italy

Abstract

This paper presents a multimodal human-robot interaction (HRI) system for educational contexts implemented on the humanoid robot Pepper. The system leverages multiple communicative channels, allowing learners to combine speech with tablet interaction while the robot responds through synchronized speech, textual captions and dynamic visual cues. To ensure robustness and reliability, we introduce the use of *Metamorphic Testing* for multimodal HRI. By validating system behavior through systematic input transformations, we demonstrate how metamorphic testing can uncover inconsistencies across linguistic, visual and cross-modal interactions. This work contributes both a novel methodological framework for evaluating multimodal HRI systems and an application to educational robotics.

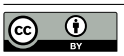
CCS Concepts

• **Computer systems organization** → **Robotics**; • **Software and its engineering** → *Software testing and debugging*.

Keywords

human robot interaction, metamorphic testing, multimodal interaction, dialogue system

ACM Reference Format:
Massimo Donini, Paolo Arcaini, Michael Oliverio, Fuyuki Ishikawa, Alessandro Mazzei, Deyun Lyu, and Cristina Gena. 2026. Coordinating Speech with Touch Input and Visual Cues in Human–Robot Interaction: A Multimodal System Evaluated through Metamorphic Testing. In *Proceedings of the 21st ACM/IEEE International Conference on Human-Robot Interaction (HRI '26), March 16–19, 2026, Edinburgh, Scotland, UK*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3757279.3785602



HRI '26, Edinburgh, Scotland, UK

ACM ISBN 979-8-4007-2128-1/2026/03
https://doi.org/10.1145/3757279.3785602

1 Introduction

Human Robot Interaction (HRI) increasingly explores how social robots can act as facilitators of learning, communication and engagement. In educational contexts, robots have been deployed as tutors, peers, or mediators to support learner-centered approaches, shifting the focus from passive knowledge transmission to active and dialogic exploration [13, 20]. Multimodality plays a central role in this process: the ability of robots to integrate speech, text, and visual cues allows them to deliver explanations that are more inclusive, adaptable, and engaging than traditional lecture-based models [10, 11, 14, 27].

This paper presents a multimodal interaction system implemented on the humanoid robot Pepper, designed to teach abstract concepts such as those in automata theory. Learners can interact with Pepper using speech and by selecting elements on its tablet, while the robot responds through synchronized speech, textual captions and dynamic visual highlights. This multimodal interaction paradigm has been shown to significantly enhance learners' engagement, attention and participation in educational contexts, particularly when combining speech with coordinated visual and embodied cues [12, 15, 28]. In addition, all of Pepper's embodiment features were kept active in their standard configuration; in particular, the robot operated in Autonomous Life mode, enabling automatic gaze behaviors and spontaneous user engagement through lifelike motions. By coordinating these communicative channels in real time, the system enables richer forms of dialogue and fosters engagement with abstract and formal topics that are typically perceived as difficult.

A central challenge in developing multimodal HRI systems is ensuring robustness and reliability across heterogeneous input and output modalities. Traditional user studies are essential for capturing human-centered dimensions of interaction, yet they are costly,

time-consuming and difficult to repeat over large input spaces. Acknowledging their fundamental role, we introduce *Metamorphic Testing* (MT) [6, 25] as an alternative or complementary method for formative evaluation in HRI. Metamorphic testing, originally developed in the field of software engineering, provides a systematic approach to address the oracle problem [4], which arises when it is difficult or even impossible to determine the correctness of a program's output using a conventional test oracle. In this context, a test oracle refers to the mechanism that decides whether the observed output of a test execution is correct. Metamorphic testing overcomes this limitation by assessing program behavior indirectly: it applies well-defined transformations to the input data and then verifies whether the resulting outputs maintain specific metamorphic relations that should logically hold [6, 25]. Recent work has shown the effectiveness of MT in domains where outputs are complex or optimality is difficult to establish, such as multi-agent path finding [33], autonomous driving systems [8, 32] and autonomous robot scheduling [16]. Moreover, its potential has been discussed in broader contexts such as automated user experience testing [7]. In addition, metamorphic relations can also be defined from the user's perspective, capturing properties that a well-designed and reliable system is expected to exhibit, independently of its implementation. This perspective extends their role beyond fault detection and verification: metamorphic relations can also serve as a tool for validation, by checking whether the system's observable behavior is consistent with user expectations [25].

In our work, we operationalize metamorphic testing for HRI by defining linguistic, visual, and cross-modal metamorphic relations. For example, paraphrased user questions should lead to semantically consistent answers, reordering elements in a diagram should not alter the underlying explanation, and speech, text, and visual highlights should remain synchronized under structural modifications. Inspired by prior applications in autonomous systems, we show how metamorphic testing can be extended to multimodal HRI, where oracles are particularly difficult to define due to the richness and variability of human interaction. Through this approach, we demonstrate how metamorphic testing can be applied to multimodal HR interactions to reveal inconsistencies, improve reliability and complement traditional evaluation methodologies. By combining methodological innovation with an educational use case, our work contributes both a novel testing framework for multimodal HRI and empirical insights into the design of interactive learning experiences with robots. In particular, in this work we experimentally answer three Research Questions (RQs) related to: RQ1) the utility of metamorphic testing in the domain of multimodal HRI, RQ2) the different performances of the multimodal (vocal/textual) channels considered in the testing, RQ3) the robustness of the system w.r.t different domain configurations (see Section 6.1 for details).

This paper is structured as follows: Section 2 reviews the state of the art; Section 3 introduces the interaction workflow and architecture of the system; Section 4 provides multimodal interaction examples; Section 5 presents the proposed metamorphic relations; Section 6 presents our application of metamorphic testing; Section 7 discusses the results and RQs; and Section 8 concludes the paper and outlines future directions.

The source code used in this work is publicly available for reproducibility purposes.[1]

2 Related work

In this section, we primarily consider a number of application contexts in which Metamorphic Testing have shown its usefulness and we subsequentially consider its applicability in the fields of HRI and user experience (UX).

2.1 Metamorphic Testing in Software Engineering

Metamorphic Testing (MT) was introduced as a strategy to address the *oracle problem*, where it is difficult or impossible to determine whether the output of a test execution is correct [6]. Instead of checking individual outputs, MT relies on *metamorphic relations* (MRs), which define necessary properties between multiple executions of a program. For example, paraphrasing a query or symmetrically modifying inputs should not change the semantic correctness of the output. Over the years, MT has been applied across diverse domains, from numerical computation to search engines [29], web services [3], machine learning [21], chess engines [19], quantum computing [1, 24], etc. The survey by Segura et al. [25] consolidates nearly two decades of work, emphasizing that the effectiveness of MT depends on the careful design of relations tailored to the program under test. They also highlight the challenge of automating the derivation of MRs and the need for formalization to ensure precision and reuse.

2.2 Metamorphic Testing in Autonomous and Multi-Agent Systems

More recently, MT has been adopted in autonomous and multi-agent contexts, where traditional oracles are difficult to define due to the scale and complexity of interactions.

Deng et al. [8] proposed RMT, a metamorphic testing approach for DNN-based perception systems of autonomous driving systems (ADSes). RMT proposes different metamorphic rules that modify the perceived images (e.g., add a pedestrian). Experiments showed that the approach is able to expose abnormal behaviors of ADSes.

Zhang et al. [33] proposed MET-MAPF, a metamorphic testing approach for multi-agent path finding algorithms. They introduced three categories of relations: environment MRs (e.g., flipping or removing obstacles in a map should not worsen path optimality), single-agent MRs (e.g., swapping initial and goal positions should preserve path feasibility), and multi-agent MRs (e.g., removing one agent should not make it harder for others to complete their tasks). Their work showed that different MRs expose different suboptimalities, demonstrating the diagnostic power of MT beyond binary success/failure checks.

Laurent et al. [16] considered autonomous delivery robots, proposing 19 MRs for testing a scheduling algorithm used in a Panasonic's delivery system. Their relations covered both order-level transformations (e.g., adding a new order should not decrease the number of completed deliveries, removing an undelivered order should not

[1]The source code is available at https://github.com/MichaelOliverio/AIMLplus-MetamorphicTesting.

affect results) and configuration-level transformations (e.g., increasing robot availability should not reduce performance). Large-scale experiments revealed that most MRs uncovered unique scheduler suboptimalities, offering engineers actionable insights into optimization trade-offs.

2.3 Towards Testing in HRI and Multimodality

In HRI, evaluation has primarily relied on empirical studies with human participants, using performance metrics, observational analysis and questionnaires. While indispensable, these methods are costly and difficult to replicate systematically. This challenge is further amplified in multimodal and embodied interaction, where the coordination of speech, affective cues, and bodily behaviors plays a central role in shaping user engagement and interaction quality [17, 26]. To address these limitations, automated testing approaches have been proposed in the neighboring field of user experience (UX). Couto et al. [7] proposed a comprehensive framework for automated UX testing, combining concepts from functional testing (e.g., coverage, reproducibility) with experiential dimensions such as usability, emotion and long-term engagement.

2.4 Gap and Research Direction

Despite advances in autonomous systems and UX testing, MT has not yet been systematically applied to either HRI or *multimodal HRI*, where robots, often equipped with a tablet, combine speech, text and visual modalities. Multimodal coordination complicates oracle construction, making these systems prime candidates for MT. Inspired by existing relations, we propose three classes of MRs tailored to multimodal HRI:

(1) **Linguistic MRs**: paraphrased or synonym-substituted user questions should yield semantically consistent robot responses, extending prior work on input transformations in NLP.
(2) **Visual MRs**: reordering or recoloring elements in SVG diagrams should not alter the explanation given by the robot.
(3) **Cross-modal MRs**: speech, textual captions, and visual highlights should remain synchronized under transformed conditions (e.g., modified layout), inspired by the scheduler relations ensuring stability across different resource configurations.

By operationalizing these relations, we extend metamorphic testing to multimodal robots, offering a scalable and repeatable methodology that complements user studies and enhances the robustness of HRI systems.

3 Interaction workflow and Architecture

This section introduces the design of the system that enables dialog-based lessons with the Pepper robot. We first outline the interaction workflow, which defines how information flows between the robot, the user and the supporting modules. We then describe the overall software architecture, shown in Figure 1. The pipeline consists of multiple processing stages. First, the system captures the user's multimodal input: the vocal request is recorded as audio files, while touch interactions on the SVG image are logged and stored as structured elements. The audio recording is then sent to a speech-to-text service, which returns a transcription in the user's native language. Both the transcribed text and the touch events

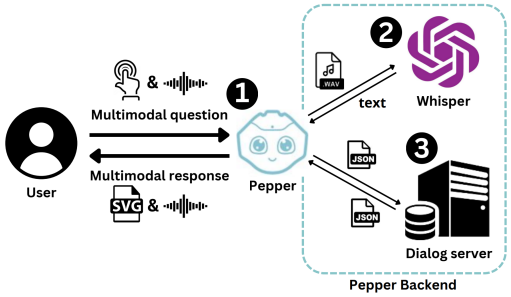


Figure 1: Information flow of the workflow pipeline.

are forwarded to the dialog server, which interprets the input and produces a JSON file encoding the robot's multimodal response. This response contains two components: the robot's spoken output and the visual modifications to be applied to the SVG image during the explanation. Finally, the Pepper robot delivers the response to the user, combining speech and visual cues to create a coherent multimodal interaction.

This workflow relies on three core components:

(1) **Pepper Robot**: Pepper displays on its tablet an SVG image representing the lesson's topic (in this case finite state automata) and waits for user queries. The robot uses its microphone to record the user's speech, while touch interactions on the tablet are logged as a list of selected elements. To improve audio quality, we implemented a custom segmentation algorithm based on a volume threshold, which detects the onset and offset of speech.
(2) **Whisper (Speech-to-Text Service)**: Audio files are sent to a Python-based application that manages the interface with the transcription system. We deployed OpenAI's Whisper[2] (medium version) on a virtual machine hosted at X (hidden for double-blind review). A dedicated REST API was implemented to receive audio recordings and return accurate transcriptions in textual format, ensuring that user queries are reliably captured in their native language.
(3) **Dialog Server**: Hosted on a separate machine and accessible via REST API, the dialog server integrates both the transcribed question and the list of touched elements. It processes the multimodal input and generates a structured JSON response. This JSON includes two components: the verbal response for Pepper to vocalize and the set of visual actions to dynamically update the SVG image (e.g., highlighting a state or transition).

Once the JSON is received, Pepper synthesizes a verbal response with the Text-To-Speech module while simultaneously applying the visual modifications to the image, so delivering a coherent multimodal explanation that combines speech with visual cues.

3.1 Dialog System

The Dialogue System employs a hybrid strategy that integrates AIML+ (Artificial Intelligence Markup Language +) [22], a declarative rule-based language inspired by the GUS (Genial Understander

[2] https://openai.com/index/whisper/

Listing 1: Example of an extended AIML+ rule for handling visual cues on a SVG images.

```
<category intent="fsa-practical" argument="transition">
    <acts>
        <act time="0">Ta:propositionalQuestion</act>
    </acts>
    <frame>
        <slot name="transitions">
            <slot-values>
                <slot-value value="q0"/>
                <slot-value value="q1"/>
                <slot-value value="?"/>
            </slot-values>
        </slot>
    </frame>
    <template>
        The transition between q0 and q1 is with value 1.
        <svgElement style-name="stroke"
        style-value="#04ed00">symbol-q0-q1</svgElement>
    </template>
</category>
```

System) paradigm [5], with the capabilities of a Large Language Model (LLM). The LLM is leveraged during the Natural Language Understanding (NLU) phase to process user utterances and extract a structured representation in the form of a frame. This frame captures the essential semantic elements of the input and is subsequently enriched with contextual information derived from the user's interactions on the tablet interface. Once the frame has been fully constructed, the interpreter analyzes its content and selects the AIML+ rule that best matches the extracted information. The use of a rule-based system is particularly suitable for our task-oriented case study, as it ensures controlled and accurate responses while mitigating the risk of hallucinations [2, 18].

AIML+ allows the definition of a set of rules forming the system's knowledge base. Each rule consists of four activators:

- **Dialogue act**: the communicative or pragmatic function of the utterance (e.g., request, confirmation) [30, 31];
- **Intent**: the goal or purpose of the user's request, representing what the user wants to achieve;
- **Argument**: a parameter or attribute that specifies additional details of the intent, providing finer granularity and context;
- **Frame**: a structured data representation consisting of *slots* that can be populated with *values* (such as strings, numbers, lists or wildcards). In our case, the frame collects the key components mentioned by the user, such as states, transitions and other elements of the automaton.

Listing 1 illustrates an example of an AIML+ rule. Following the approach of Oliverio et al. [23], we implemented an automated pipeline that generates AIML+ rules directly from automata encoded as SVG images. We created nine distinct automata configurations, with every graphical element annotated by a unique identifier. The extraction module analyses the structure and hierarchy of the SVG document. It parses element types and attributes, interprets nesting relationships and maps these semantic and structural cues onto AIML+ rule templates. By inspecting element ids, grouping,

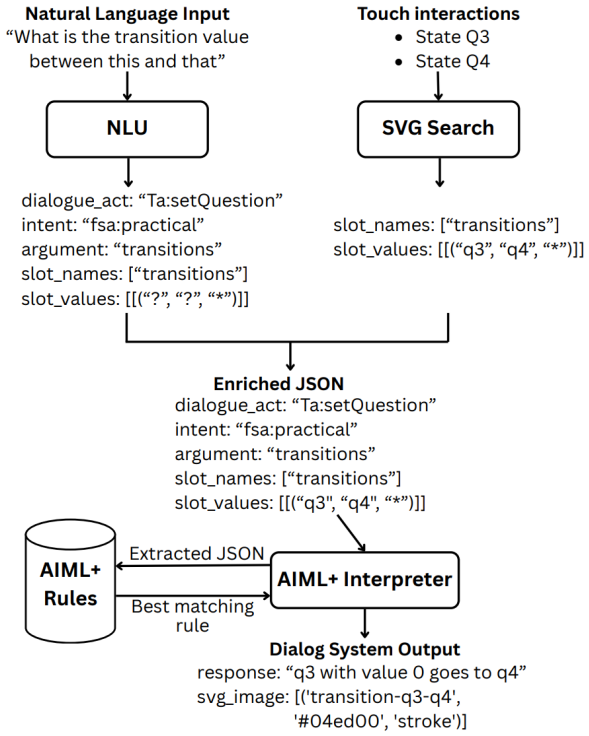


Figure 2: Overview of the dialog system pipeline from user input to system output.

geometric primitives and textual content, the system automatically instantiates rules that capture state definitions, transition conditions and contextual relationships. Figure 2 shows an example of the dialog system workflow, illustrating the process from multimodal user input to multimodal system response.

The NLU module is responsible for extracting the four activators that define AIML+ rules, which are subsequently used to identify the most relevant rule by the interpreter. To develop this module, we relied on the NoVAGraphS corpus [22], which was divided into training, validation, and test sets following an 80/10/10 split. On this dataset, we fine-tuned small-scale LLMs to perform the text-to-JSON transformation task. Initially, we fine-tuned Llama-3.2-1B-Instruct[3], which achieved unsatisfactory performance (weighted F1 score of 0.383 averaged over all labels in the extracted JSON). Subsequently, we fine-tuned Llama-3.2-3B-Instruct[4], obtaining significantly better results (weighted F1 score of 0.794 averaged over all labels). The models were trained on an Nvidia RTX4050 GPU using QLoRA [9] (4-bit quantization), with 4 epochs and a batch size of 4. The JSON produced by the NLU module is then validated and enriched with touch input information. The touch component provides the IDs of the SVG elements selected by the user, which are retrieved to gather additional data and augment the extracted frame. Finally, the dialog manager receives the validated JSON and forwards it to the interpreter, which selects the most appropriate AIML+ rule. Rule selection is based on matching the intent and

[3] https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct
[4] https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct

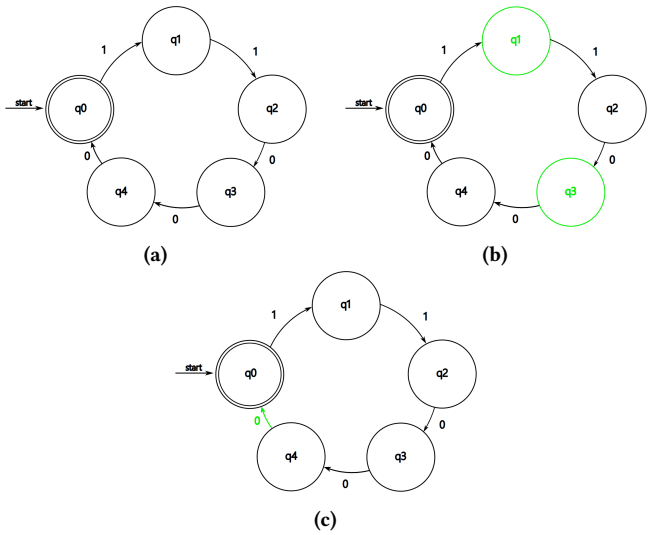


Figure 3: Pepper robot's tablet before and while communicating the sentence:"There is no transition between q1 and q3." and "The transition from q4 to q0 has value 0."

argument, as well as the similarity between the enriched extracted and the frames defined in AIML+ rules.

4 Multimodal Interaction Examples

This section presents two examples of multimodal interactions with the developed system.

In the first example (Figure 3a), the user asks the robot, "Is there a transition between this state and that one?" while selecting states *q1* and *q3* on the tablet. The dialog system returns the textual answer "There is no transition between *q1* and *q3*." Pepper then speaks this sentence to the user and, simultaneously, the tablet interface highlights on the tablet the key elements mentioned in the visual part of the response. Specifically, the system visually emphasizes states *q1* and *q3*, as shown in Figure 3b.

In the second example (taking Figure 3a again), the user asks the robot, "How do I reach the final state from this?" while selecting state *q4* on the tablet. The dialog system returns the textual answer "The transition from *q4* to *q0* has value *0*." Pepper then speaks this sentence to the user and, simultaneously, the tablet interface highlights on the tablet the key elements mentioned in the visual part of the response. Specifically, the system visually emphasizes the transition from *q4* to *q0* and its value, as shown in Figure 3c.

5 Metamorphic Testing of the Dialog System

The core component of the system under evaluation, and the focus of our forthcoming tests, is the Dialog System, which manages multimodal communication both for the user and the robot. As previously discussed, this system leverages an LLM for its Natural Language Understanding (NLU) module. While this choice enhances flexibility, it also raises concerns regarding reliability and robustness. An extensive testing phase is so required to assess these aspects. A major challenge in testing such systems lies in the oracle problem [4]: given the responses obtained from the dialogue system, it is not always clear whether these should be considered optimal or not and metamorphic testing has often been applied in domains affected by this problem [34]. In this work, we propose its application to evaluate the quality and robustness of this system.

Metamorphic testing relies on a set of MRs. Each metamorphic relation M is defined by two components:

(1) **Transformation** ($M.\text{trans}(t_{\text{src}})$): given a source test case t_{src}, this function generates a set of follow-up test cases $\{t_1^{\text{fol}}, \ldots, t_k^{\text{fol}}\}$. In some cases, only one follow-up test case is produced; in others, the transformation may not be applicable, in which case it returns the empty set $\emptyset$.
(2) **Relation** ($M.\text{rel}(t_{\text{src}}, t_{\text{fol}})$): this defines an expected property between the results $Res(t_{\text{src}})$ of the source test case and the results $Res(t_{\text{fol}})$ of a follow-up test case. If the property holds, the system is considered to behave as expected; otherwise, a violation is detected. In the general case of multiple follow-up tests $\{t_1^{\text{fol}}, \ldots, t_k^{\text{fol}}\}$, the property is satisfied only if it holds for all of them. By definition, the property is also satisfied when $M.\text{trans}(t_{\text{src}}) = \emptyset$, i.e., when no follow-up test is produced.

In our system, a test case t is given by a multimodal user input, which integrates the transcription of a spoken utterance with the interaction performed on the robot's tablet, together with the specific SVG image displayed by Pepper. The output is the multimodal response of the robot defined by the textual reply and the corresponding set of visual modifications to apply to the SVG image.

The outputs of the dialog system are compared between the source test case and each of its follow-up test cases. In this work, the MRs are defined in terms of two types of expected properties:

- **Same Response:** This property requires that the dialog system generates identical outputs for both the source test case and its follow-up. A violation indicates that the system produced inconsistent responses to inputs that should have been equivalent.
- **Adaptive Response:** This property requires that the dialog system produces distinct outputs for the source test case and its follow-up. A violation occurs when the system fails to adjust its response according to the transformation applied. For example, if a follow-up test removes a state or transition from the multimodal input, the system should also remove the corresponding element from its response.

Note that, in our case, a violation of an MR does not necessarily imply an incorrect response. Rather, it may indicate that the response does not fully satisfy or address the intended request.

5.1 Proposed Metamorphic Relations

This section introduces the MRs defined to evaluate the robustness and reliability of the dialog system. We distinguish between two main categories of MRs, depending on which component of the source test case they transform: those that modify the multimodal user input and those that modify the SVG image that the system is required to explain.

5.1.1 *MRs affecting multimodal user input.*

Add Filler Words (AFW): it generates follow-up test cases by

inserting filler words into the transcription of the user's request (e.g., eh, uhm, so). This MR creates multiple follow-up cases, ranging from the insertion of a single filler word to the addition of a filler word after every word in the original test case, to assess system robustness under extreme conditions. The expected property is that the system's response to these follow-up tests should remain unchanged. Example of source and follow-up test case:

```
Source:  "What is the final state of an automaton?"
Follow-up: "Mmm what is mmm the mmm final state of
            an uhm automaton?"
```

Sentence Inversion/Anastrophe (SIA): it generates follow-up test cases by transforming the user's request through word-order inversions or the application of anastrophe. The expected property is that the system's response to the follow-up tests should remain unchanged. Example of source and follow-up test cases:

```
Source:  "How many arcs marked by 1 are there?"
Follow-up: "There are how many arcs marked by 1?"
```

Passive/Active Sentence (PAS): it generates follow-up test cases by transforming the transcription of the user's request between active and passive voice, depending on the form of the original sentence. The expected property is that the system's response to these follow-up tests should remain unchanged. Example of source and follow-up test case:

```
Source:  "How do you represent the automaton?"
Follow-up: "How is the automaton represented?"
```

Replace With Synonyms (RWS): it generates follow-up test cases by transforming the transcription of the user's request through the replacement of words with their synonyms. The expected property is that the system's response to these follow-up tests should remain unchanged. Example of source and follow-up test case:

```
Source:  "Is there a transition between q5 and q7?"
Follow-up: "Is there an arc between q5 and q7?"
```

Word-Level Errors (WLE): it generates follow-up test cases by introducing word-level errors into the transcription of the user's request, such as changing singular to plural (or vice versa), removing double letters or applying other grammatical modifications. These transformations are intended to simulate potential user pronunciation errors. The expected property is that the system's response to these follow-up tests should remain unchanged. Example of source and follow-up test case:

```
Source:  "Is there a particular pattern among the arcs?"
Follow-up:"Is there a particular patern among the arcs?"
```

Multiple User Requests (MUR): it generates follow-up test cases by combining multiple user requests into a single one. The expected property is that the system's response to these follow-up tests should correspond to the composition of the responses that would have been produced if each request had been submitted individually. Example of source and follow-up test case:

```
Source:  "What are the states connected to q3?"
Follow-up: "What are the states connected to q3? How
            many transitions does the automata have?"
```

Touch Input Request (TIR): it tests explanations for a single element (e.g., a state, a transition, etc.). It generates follow-up test cases by replacing the element for which an explanation is requested with the corresponding touch input on the tablet for that element. The expected property is that the system's response to these follow-up tests should remain unchanged. For instance, given the following source test case:

```
"user_input": "Tell me about the transition
    value between q3 and q4?",
"ids": []
```

a possible follow-up test case is:

```
"user_input": "",
"ids": ["value-q3-q4"]
```

Multimodal Input Request (MIR): it generates follow-up test cases by modifying the user's request, replacing references to elements of the SVG image, such as states or transitions, with pronouns (e.g., this, that). At the same time, the replaced elements are added to the list of interactions performed on the tablet. The expected property is that the system's response to the follow-up tests should not change. For instance, given the following source test case:

```
"user_input": "Is there a connection between the
    states Q1 and Q2?",
"ids": []
```

a possible follow-up test case is:

```
"user_input": "Is there a connection between
    this and that?",
"ids": ["Q1", "Q2"]
```

5.1.2 MRs affecting SVG image.

Add Probe State (APS): it modifies the structure of the automaton by generating follow-up test cases in which a probe state (i.e., a state without incoming or outgoing transitions) is added to the SVG image. The expected property is that the system's response to queries concerning state transitions and state reachability should remain unchanged.

Add/Remove State (ARS): it modifies the structure of the automaton by generating follow-up test cases in which a state in the SVG image is either added or removed. In particular, the transformation considers two states that were originally connected to each other and reconnects them through the newly added state (or restores the direct connection if the state is removed). The expected property is that the system's response to queries concerning the number of states and their connections should adapt consistently to the modified structure.

Add/Remove Transition (ART): it modifies the structure of the automaton by generating follow-up test cases in which a transition between two states in the SVG image is either added or removed. The expected property is that the system's response to queries regarding state transitions and state reachability should adapt consistently to the modified structure.

Change Transition Value (CTV): it modifies the structure of the automaton by generating follow-up test cases in which the values associated with state transitions are altered. The expected property

Table 1: Number of source test cases of each MR and number of generated follow-up tests

	AFW	SIA	PAS	RWS	WLE	MUR	TIR	MIR	APS	ARS	ART	CTV	CSE
# source test cases	485	433	295	251	123	485	34	54	190	210	380	246	198
# follow-up test cases	2937	578	295	818	483	688	34	54	190	210	380	246	198

is that the system's response to queries concerning state transitions should adapt consistently to the modified values.

Change Start/End (CSE): it modifies the structure of the automaton by generating follow-up test cases in which the start state or the goal state is changed. The expected property is that the system's response to queries concerning the automaton's structure should adapt consistently to the modified configuration.

6 Experiment Design

This section describes the experiments we designed to assess the robustness and reliability of the dialog system.

6.1 Research Questions

To evaluate the performance of the multimodal dialogue system, we pose the following research questions (RQs):

RQ1 *What is the utility of the proposed metamorphic testing approach?*
This research question evaluates how frequently the different MRs are violated, i.e., cases in which the system produces incorrect or partially relevant responses. Such violations highlight issues that should be carefully inspected before deploying the system in real-world scenarios.

RQ2 *Is there a performance difference between the two components (textual/visual) of the system's multimodal responses?*
This research question investigates whether MR violations occur more frequently in the textual component or in the visual cues of the response, in order to identify which aspect requires greater attention for future improvements.

RQ3 *Does the system remain robust and correctly adapt its responses to new automaton configurations?*
This research question investigates the extent to which the system is flexible in providing explanations for different automaton structures that were not initially anticipated.

In addition to addressing the above RQs, we perform a qualitative analysis aimed at characterizing the types of failures revealed by the tests. This supplementary analysis provides further insights into the nature of the system's weaknesses and helps identify areas that require refinement before deployment.

6.2 Experimental Settings

To evaluate the optimality of the dialog system, it is necessary to subject it to a set of tests that are representative of real human–robot interactions and that cover a wide variety of possible scenarios. For this reason, the source test cases were derived from dialogues that took place between real users and previous versions of the dialog system. This corpus consists of the transcription of 485 utterances recorded during those interactions.

Table 1 reports statistics on the transformations applied by each MR. In particular, the column *# source test cases* indicates how many times a given MR was applied, while the column *# follow-up test cases* shows the total number of generated followup test cases. As can be observed, not all 485 sentences in the corpus could be used for every MR; for instance, some sentences cannot be transformed from active to passive form, or vice versa.

Table 2: Violation Rates (VR_{MR}) for each Metamorphic Relation (MR)

MR	VR_{MR} %	
	VR^{text}_{MR}	VR^{visual}_{MR}
Add Filler Words (AFW)	20.8	13.6
Sentence Inversion/Anastrophe (SIA)	16.9	10.5
Passive/Active Sentence (PAS)	17.7	12.0
Replace With Synonyms (RWS)	28.4	18.8
Word-Level Errors (WLE)	11.3	8.9
Multiple User Requests (MUR)	100	49.7
Touch Input Request (TIR)	5.8	2.9
Multimodal Input Request (MIR)	5.4	5.4
Add Probe State (APS)	9.4	6.8
Add/Remove State (ARS)	7.1	7.1
Add/Remove Transition (ART)	8.9	6.0
Change Transition Value (CTV)	8.1	4.0
Change Start/End (CSE)	2.0	1.0

6.3 Evaluation Metrics

In order to address the proposed RQs, we report, for each MR, the violation rate VR_{MR}, i.e., the percentage of times a specific MR is violated out of the total number of times it is applicable. In particular, with respect to RQ2, VR_{MR} is reported separately for the two components of the multimodal response: VR^{text}_{MR} for the textual output, and VR^{visual}_{MR} for the visual cues.

7 Experiment Results

This section presents the results of the experiments described above and addresses the previously stated research questions. Table 2 summarizes the VR_{MR} observed for all the proposed MRs.

Answering RQ1: As shown in Table 2, the proposed tests are able to detect violations, thereby demonstrating their capability to expose suboptimal behaviors and robustness weaknesses. However, different MRs exhibit different violation rates. In general, MRs that modify the user input tend to fail more frequently than those that alter the structure of the SVG image to be explained. This result can be interpreted as an indicator of system safety: the model is not prone to providing misleading explanations to the user. When the inputs are modified or ambiguous, the system may struggles

to fully satisfy the request, leading to incomplete or less precise responses rather than incorrect explanations. Conversely, when the SVG structure is altered but the user input remains clear, the system shows greater robustness, as it is able to correctly retrieve and describe the relevant information from the SVG.

The MR with the highest failure rate is the MUR, which evaluates system responses when the user issues multiple requests simultaneously. This MR highlights a limitation of the system, which is currently designed to handle only one request at a time. Nevertheless, even in such cases, the system usually manages to correctly answer at least one of the two requests. Indeed, when comparing the follow-up responses with the individual requests, the system fails to provide a correct answer to both in only 8.1% of the cases. This finding further indicates that, while the system may often produce partial or incomplete explanations in failure cases, it rarely provides completely incorrect ones with respect to the user's request.

Among the MRs that modify user input, those with the lowest violation rates were the ones about multimodal requests, namely the Touch Input Request and Multimodal Input Request. This is a highly positive outcome, highlighting the importance and effectiveness of multimodal interaction, which fully leverages all available communication channels. Since multimodal requests combine textual input with user touch interactions, they are inherently more concrete, as they refer to elements explicitly present in the SVG image being touched. Such concreteness makes these requests easier for the system to handle compared to more abstract queries, while also reducing the likelihood of erroneous requests concerning elements not actually present in the image.

Answering RQ2: Table 2 shows that this is indeed the case: for all MRs, the violation rate is higher for the textual component than for the visual cues. This result is expected, as more abstract requests regarding general automata properties do not trigger multimodal outputs or highlightable elements in the SVG, leaving only the textual response to be evaluated. In other cases, textual answers may be partially inconsistent or incomplete—for instance, when asked about the value of a transition, the system may state that a transition exists between two states instead of providing its value. However, in such situations the visual cues remain correct, as they still highlight the relevant transition. This illustrates the value of providing explanations through multiple output channels: while the textual channel may carry richer information, visual cues often deliver more reliable content and can guide the user's attention to the relevant elements, sometimes sufficiently addressing the information need even when the textual response is imperfect.

Answering RQ3: As already observed when discussing RQ1, the MRs with the lowest violation rates are those involving transformations of the automaton structure. This indicates good flexibility and adaptability of the dialogue system in providing correct answers even when confronted with automata not seen during its initial design. Such robustness is particularly relevant, as it makes the system suitable for use in diverse contexts and prevents repetitive interactions. Moreover, it allows the system to be employed across different stages of learning, from simple and elementary configurations to more complex ones.

Qualitative evaluation: From a more qualitative perspective, the results appear consistent with expectations and aligned with the characteristics of the dialogue system. Many of the observed violations can likely be attributed to the NLU phase, which in this version was handled by Llama 3.2 with 3B parameters. It is reasonable to assume that employing larger LLMs for this component could reduce the number of violations. Furthermore, in our evaluation framework, any deviation from the expected output was considered an error, even when the response was only partially incorrect or not fully aligned with the request. This choice reflects the dual role of metamorphic relations, which are not only suitable for detecting faults in the software under test (verification) but also for assessing whether the system behaves in accordance with user expectations (validation) [25]. While maintaining consistent and predictable interactions is crucial for an optimal user experience, it is worth noting that, despite some MRs reporting relatively high violation rates, the overall quality of the system's responses remained strong. The following examples illustrate cases in which the system generates answers that are not fully consistent with the corresponding questions. For the query *"What happens when a 0 comes to state q1?"*, instead of stating that state *q1* has no outgoing transitions labeled with 0, the system replies *"The transition between q1 and q2 is with value 1."* Although the answer is incorrect, it still remains topically related to the question. For the query *"What is the final state of an automaton?"*, the system provides the response *"The final state is q0."* Again, the answer is inaccurate, but it is semantically close to the intended concept. In general, when the system encounters a request that it cannot process or understand, it resorts to a fallback response such as *"Could you be more specific?"*.

8 Conclusions

This work presented a multimodal human–robot interaction system implemented on the humanoid robot Pepper. The system leverages multiple communicative channels, using speech and touch for user input and speech and visual cues for robot output. It was designed to deliver lessons explaining concepts that can be represented through structured images such as SVGs; in this study, the chosen domain was finite state automata. To evaluate the robustness and reliability of the system prior to deploying it in real settings, we conducted a set of tests based on MRs. These tests revealed violations across the proposed MRs and demonstrated their utility in identifying key aspects of the system that require improvement for future implementations before its deployment in real-world scenarios.
As future work, we plan to conduct a thorough embodiment study to analyze the impact of the robot's physical presence and multimodal behaviors on user interaction. Additionally, we aim to compare metamorphic testing with standard methodologies for multimodal robotic systems in order to better assess its effectiveness.

Acknowledgments

M. Donini worked under a cooperative agreement between the University of Turin and Intesa Sanpaolo Innovation Center.
P. Arcaini and F. Ishikawa are supported by Engineerable AI Techniques for Practical Applications of High-Quality Machine Learning-based Systems Project (Grant Number JPMJMI20B8), JST-Mirai. P. Arcaini is also supported by the ASPIRE grant No. JPMJAP2301, JST.

References

[1] Rui Abreu, João Paulo Fernandes, Luis Llana, and Guilherme Tavares. 2023. Metamorphic Testing of Oracle Quantum Programs. In *Proceedings of the 3rd International Workshop on Quantum Software Engineering* (Pittsburgh, Pennsylvania) *(Q-SE '22)*. Association for Computing Machinery, New York, NY, USA, 16–23. doi:10.1145/3528230.3529189

[2] Adel Abusitta, Miles Q. Li, and Benjamin C.M. Fung. 2024. Survey on Explainable AI: Techniques, challenges and open issues. *Expert Syst. Appl.* 255, PC (Dec. 2024), 18 pages. doi:10.1016/j.eswa.2024.124710

[3] John Ahlgren, Maria Eugenia Berezin, Kinga Bojarczuk, Elena Dulskyte, Inna Dvortsova, Johann George, Natalija Gucevska, Mark Harman, Maria Lomeli, Erik Meijer, Silvia Sapora, and Justin Spahr-Summers. 2021. Testing Web Enabled Simulation at Scale Using Metamorphic Testing. In *Proceedings of the 43rd International Conference on Software Engineering: Software Engineering in Practice* (Virtual Event, Spain) *(ICSE-SEIP '21)*. IEEE Press, Piscataway, NJ, USA, 140–149. doi:10.1109/ICSE-SEIP52600.2021.00023

[4] Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. 2015. The Oracle Problem in Software Testing: A Survey. *IEEE Trans. Softw. Eng.* 41, 5 (May 2015), 507–525. doi:10.1109/TSE.2014.2372785

[5] Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry Thompson, and Terry Winograd. 1977. GUS, a frame-driven dialog system. *Artif. Intell.* 8, 2 (April 1977), 155–173. doi:10.1016/0004-3702(77)90018-2

[6] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, T. H. Tse, and Zhi Quan Zhou. 2018. Metamorphic Testing: A Review of Challenges and Opportunities. *ACM Comput. Surv.* 51, 1, Article 4 (Jan. 2018), 27 pages. doi:10.1145/3143561

[7] Marta Couto, Rui Prada, Pedro M. Fernandes, Manuel Lopes, Davide Prandi, I.S. Wishnu B. Prasetya, and Saba Gholizadeh Ansari. 2025. Towards a comprehensive framework for automated testing of user experience. *Quality and User Experience* 10, 3 (2025), 1–22. doi:10.1007/s41233-025-00073-6

[8] Yao Deng, Xi Zheng, Tianyi Zhang, Huai Liu, Guannan Lou, Miryung Kim, and Tsong Yueh Chen. 2023. A Declarative Metamorphic Testing Framework for Autonomous Driving. *IEEE Transactions on Software Engineering* 49, 4 (2023), 1964–1982. doi:10.1109/TSE.2022.3206427

[9] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. arXiv:2305.14314 [cs.LG]

[10] Massimo Donini, Cristina Gena, and Alessandro Mazzei. 2024. Multimodal Strategies for Robot-to-Human Communication. In *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, New York, NY, USA, 417–421.

[11] Massimo Donini, Michael Oliverio, Pier Felice Balestrucci, Luca Anselma, Cristina Gena, Alessandro Mazzei, Matteo Nazzario, and Irene Borgini. 2025. Towards a Structured Multimodal Speech-Image Coordination. In *Adjunct Proceedings of the 33rd ACM Conference on User Modeling, Adaptation and Personalization (UMAP)*. ACM, New York, NY, USA. doi:10.1145/3708319.3733662

[12] Ka Yan Fung, Kwong Chiu Fung, Tze Leung Rick Lui, Kuen Fung Sin, Lik Hang Lee, Huamin Qu, and Shenghui Song. 2025. Exploring the impact of robot interaction on learning engagement: a comparative study of two multi-modal robots. *Smart Learning Environments* 12, 1 (2025), 12.

[13] Cristina Gena, Claudio Mattutino, Gianluca Perosino, Massimo Trainito, Chiara Vaudano, and Davide Cellie. 2020. Design and development of a social, educational and affective robot. In *2020 IEEE conference on evolving and adaptive intelligent systems (EAIS)*. IEEE, IEEE, Piscataway, NJ, USA, 1–8.

[14] Chih-Chien Hu, Yu-Fen Yang, and Nian-Shing Chen. 2022. Human–robot interface design – the 'Robot with a Tablet' or 'Robot only', which one is better? *Behaviour and Information Technology* 42 (07 2022), 1–14. doi:10.1080/0144929X.2022.2093271

[15] An Jianliang. 2024. A multimodal educational robots driven via dynamic attention. *Frontiers in Neurorobotics* 18 (2024), 1453061.

[16] Thomas Laurent, Paolo Arcaini, Xiao-Yi Zhang, and Fuyuki Ishikawa. 2024. Metamorphic Testing of an Autonomous Delivery Robots Scheduler. In *2024 IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, Piscataway, NJ, USA, 361–372. doi:10.1109/ICST60714.2024.00040

[17] Xiaofeng Liu, Qincheng Lv, Jie Li, Siyang Song, and Angelo Cangelosi. 2024. Multimodal emotion fusion mechanism and empathetic responses in companion robots. *IEEE Transactions on Cognitive and Developmental Systems* 16 (2024).

[18] Michael McTear. 2022. *Conversational AI: Dialogue systems, conversational agents, and chatbots*. Springer Nature, Heidelberg, Germany.

[19] Manuel Méndez, Miguel Benito-Parejo, Alfredo Ibias, and Manuel Núñez. 2023. Metamorphic testing of chess engines. *Information and Software Technology* 162 (2023), 107263. doi:10.1016/j.infsof.2023.107263

[20] Mayiana Mitevska et al. 2025. Integration of Social Robots in the Educational Environment... https://api.semanticscholar.org/CorpusID:280517671

[21] Prudhviraj Naidu, Hemanth Gudaparthi, and Nan Niu. 2021. Metamorphic Testing for Convolutional Neural Networks: Relations over Image Classification. In *2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI)*. IEEE Press, Piscataway, NJ, USA, 99–106. doi:10.1109/IRI51335.2021.00020

[22] Michael Oliverio, Pier Felice Balestrucci, Luca Anselma, and Alessandro Mazzei. 2025. AIML+: Enhancing AIML for the Educational Domain Through Frames and Large Language Models. In *Artificial Intelligence in Education: 26th International Conference, AIED 2025, Palermo, Italy, July 22–26, 2025, Proceedings, Part IV* (Palermo, Italy). Springer-Verlag, Berlin, Heidelberg, 176–189. doi:10.1007/978-3-031-98459-4_13

[23] Michael Oliverio, Margherita Piroi, Daniele De Giorgi, Pier Felice Balestrucci, Carola Manolino, Alessandro Mazzei, Luca Anselma, Cristian Bernareggi, Marina Serio, Cristina Sabena, Tiziana Armano, Sandro Coriasco, and Anna Capietto. 2024. NoVAGraphS: Towards an Accessible Educational-Oriented Dialogue System. In *Proceedings of the Second International Workshop on Artificial INtelligent Systems in Education co-located with 23rd International Conference of the Italian Association for Artificial Intelligence (AIxIA 2024)*. CEUR-WS.org, Aachen, Germany.

[24] Matteo Paltenghi and Michael Pradel. 2023. MorphQ: Metamorphic Testing of the Qiskit Quantum Computing Platform. In *Proceedings of the 45th International Conference on Software Engineering* (Melbourne, Victoria, Australia) *(ICSE '23)*. IEEE Press, Piscataway, NJ, USA, 2413–2424. doi:10.1109/ICSE48619.2023.00202

[25] Sergio Segura, Gordon Fraser, Ana B. Sanchez, and Antonio Ruiz-Cortes. 2016. A Survey on Metamorphic Testing. *IEEE Transactions on Software Engineering* 42, 9 (2016), 805–824. doi:10.1109/TSE.2016.2532875

[26] Hang Su, Wen Qi, Jiahao Chen, Chenguang Yang, Juan Sandoval, and Med Amine Laribi. 2023. Recent advancements in multimodal human–robot interaction. *Frontiers in Neurorobotics* 17 (2023), 1084000.

[27] Qingfeng Sun, Yujing Wang, Can Xu, Kai Zheng, Yaming Yang, Huang Hu, Fei Xu, Jessica Zhang, Xiubo Geng, and Daxin Jiang. 2021. Multimodal dialogue response generation. arXiv:2110.08515 [cs.CL] https://arxiv.org/abs/2110.08515

[28] Yvonne Vezzoli. 2019. *Exploring the opportunities of multimodal literacies for the participation and learning of young people with dyslexia in multimodal digital environments: informing learning design*. Ph.D. thesis. Università Ca' Foscari Venezia. https://iris.unive.it/handle/10579/15004

[29] Xinyi Wang, Gaolei Yi, and Yichen Wang. 2021. Automated Functional Testing of Search Engines using Metamorphic Testing. In *2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, Piscataway, NJ, USA, 22–29. doi:10.1109/QRS-C55045.2021.00014

[30] Alan R. White. 1963. How to Do Things with Words. *Analysis* 23 (1963), 58–64. http://www.jstor.org/stable/3326622

[31] Ludwig Wittgenstein. 1953. *Philosophical Investigations. Philosophische Untersuchungen*. Macmillan, New York, NY, USA. https://psycnet.apa.org/record/1954-01801-000

[32] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 2018. DeepRoad: GAN-Based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering* (Montpellier, France) *(ASE 2018)*. Association for Computing Machinery, New York, NY, USA, 132–142. doi:10.1145/3238147.3238187

[33] Xiao-Yi Zhang, Yang Liu, Paolo Arcaini, Mingyue Jiang, and Zheng Zheng. 2024. MET-MAPF: A Metamorphic Testing Approach for Multi-Agent Path Finding Algorithms. *ACM Trans. Softw. Eng. Methodol.* 33, 8, Article 198 (Nov. 2024), 37 pages. doi:10.1145/3669663

[34] Zhi Quan Zhou, Liqun Sun, Tsong Yueh Chen, and Dave Towey. 2018. Metamorphic relations for enhancing system understanding and use. *IEEE Transactions on Software Engineering* 46, 10 (2018), 1120–1154.

Received 2025-09-30; accepted 2025-12-01