Project report for the course Multisensory Interactive Systems
Academic Year: 2025-2026

MazeTilt

Nicola Cappellaro - nicola.cappellaro@studenti.unitn.it - 256119
Riccardo Zannoni - riccardo.zannoni@studenti.unitn.it - 256121

Abstract

This report presents the design, implementation, and evaluation of MazeTilt, a multisensory interactive virtual maze developed for the Multisensory Interactive Systems course at the University of Trento.
The system allows users to control a virtual ball by physically tilting a custom-built handheld controller equipped with an accelerometer, directly mapping the device orientation to the maze inclination and the ball's motion.
The interaction is enhanced through multimodal feedback, combining visual rendering, real-time auditory cues generated via Pure Data, and vibrotactile feedback provided by ERM motors. Continuous rolling sounds are mapped to the ball's velocity, discrete sounds signal collisions and falls, and vibrotactile feedback conveys proximity to hazards and impacts.
The project investigates whether the integration of auditory and haptic feedback can improve task performance and perceived immersion compared to a visual-only interaction. A controlled within-subject user study was designed following established experimental guidelines, collecting both objective performance metrics (completion time, errors) and subjective measures of usability, immersion, and emotional response.
Results from evaluation suggest that multimodal feedback increases user awareness of the environment, reduces errors, and enhances perceived immersion, indicating the potential benefits of multisensory interaction in interactive entertainment scenarios and motor control tasks.

1 Introduction

The project focuses on the development of a multisensory interactive virtual maze, in which users control a ball by physically tilting an accelerometer-based controller, hereafter referred to as the Gamepad. The objective of the task is to reach a goal area while avoiding holes within the maze. Tilt-based interaction mimics real-world physical manipulation, making the task intuitive while still challenging. The main hypothesis driving the project is that augmenting a visual interface with auditory and haptic feedback improves user performance, immersion, and perceived control compared to visual feedback alone.
The project is situated within the context of multisensory interactive systems and embodied interaction, and the system is conceived as an experimental platform for studying sensorimotor interaction in an interactive entertainment scenario.
In particular, the project builds on theoretical perspectives introduced during the course, such as

- Multisensory integration, where information from different sensory modalities is combined to enhance perception and action, and
- Action-perception coupling, in which users' movements directly produce immediate perceptual consequences through visual, auditory, and haptic feedback.

2 Related work

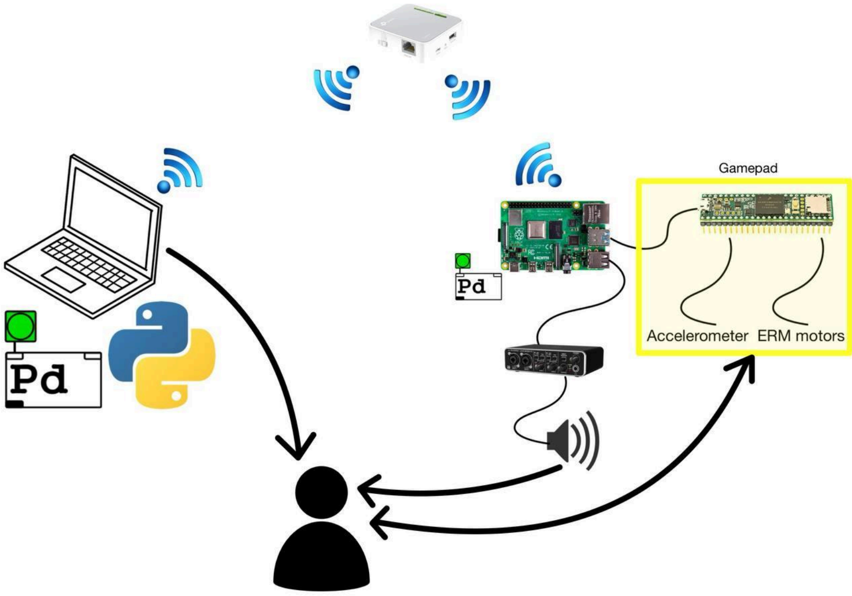
Several interactive systems have explored tilt-based interaction, particularly in mobile games and rehabilitation tools.
Previous research in multisensory interaction shows that:

- Auditory feedback can enhance spatial awareness and timing.
- Vibrotactile feedback improves motor precision and error awareness.
- Multimodal systems outperform unimodal ones in task performance and engagement.

Unlike most tilt-based smartphone games, where control and visualization are integrated within the same device, our system separates the input interface from the graphical display. In fact, In this system, input commands are generated through direct physical tilting of the gamepad, unlike most related mobile applications where interaction is primarily based on on-screen software buttons.
This design can be interpreted as an example of a Tangible User Interface (TUI), where physical manipulation of a tangible object is directly mapped to digital actions, fostering a more embodied and engaging interaction.

3 Architecture design

The system is an interactive tilt-controlled labyrinth game that integrates visual, auditory, and vibrotactile feedback. A custom-built gamepad acts as the physical controller, while a PC handles real-time rendering, physics simulation, and audio generation. The architecture follows a distributed design, where sensing and haptic actuation are handled by a microcontroller, and computation-intensive tasks are performed on the PC.

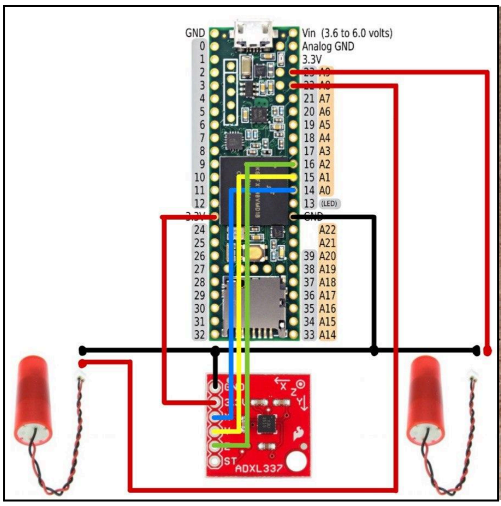


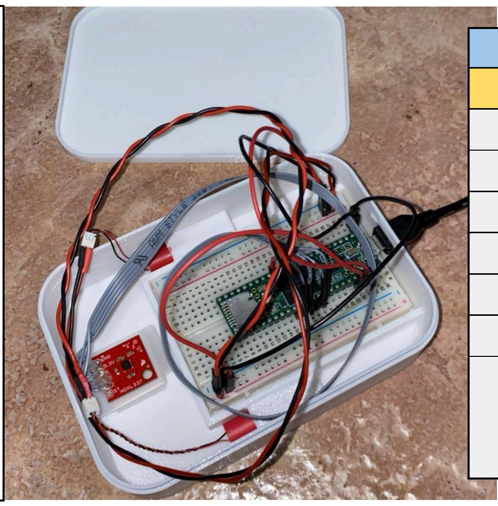
Hardware Architecture

Our hardware setup includes a router used to create a local network. Both the PC and the Raspberry Pi are connected to the router. The gamepad controller and the sound card with loudspeaker are directly connected to the Raspberry Pi.
The gamepad contains:

- A Teensy 3.6 microcontroller mounted on a breadboard
- An ADXL337 analog accelerometer used to measure tilt along multiple axes
- 2x ERM vibration motors providing vibrotactile feedback

The accelerometer continuously measures the orientation of the gamepad. These values are sent to the PC via OSC communication with Pure Data patch, where they are mapped to the virtual inclination of the labyrinth. The ERM motors are directly controlled by the microcontroller based on commands received from the PC.





Teensy 3.6	
pin	desc
3.3V	Accel 3.3V
A0	accel - x
A1	accel - y
A2	accel - z
A8	ERM motor 1
A9	ERM motor 2
GND	accel - gnd
	ERM motor 1
	ERM motor 2

Software Architecture

Rendering and Physics

On the PC, a Python application manages:

- Rendering of the labyrinth and ball using Pygame and OpenGL

- Physics simulation, where gravity is dynamically adjusted according to the tilt values received from the controller

Tilting the physical gamepad results in a corresponding tilt of the virtual labyrinth, causing the ball to move under simulated gravity.

Audio Feedback

Audio feedback is generated using one Pure Data patch, where each block is responsible for a specific game event:

- Rolling sound proportional to the ball's velocity
- Bounce sound triggered by collisions with walls
- Explosion sound when the ball falls into a hole
- Victory sound when the goal is reached

Audio communication between the Python application and Pure Data is handled through OSC (Open Sound Control) messages, enabling low-latency and event-based sound synthesis.

Vibrotactile Feedback

Vibrotactile feedback is used to convey proximity to hazards:

- When the ball approaches a hole, vibration intensity increases proportionally to the distance
- This feedback is sent from the PC via OSC to the Pure Data patch and then sent to the microcontroller via serial commands
- The microcontroller drives the ERM motors accordingly, allowing the user to physically perceive danger without relying solely on visual cues

Communication Architecture

The system uses two communication channels:

- Serial communication between Pure Data patches and the microcontroller for accelerometer data and haptic commands
- OSC communication between the Python application and Pure Data patches for audio control and haptic commands

Both channels are assumed to be sufficiently reliable and low-latency to support real-time interaction.

Assumptions and Justification

System Assumptions

- Serial and OSC communication are reliable and introduce negligible latency
- The PC can handle real-time graphics rendering and audio synthesis simultaneously
- At system startup, the gamepad is placed flat on a horizontal surface to allow proper accelerometer calibration

The calibration assumption is reasonable, as requiring an initial neutral position is common practice in tilt-based interfaces and ensures accurate mapping between physical orientation and virtual inclination.

User Assumptions

- Users can intuitively understand and use a tilt-based interface
- Users can perceive ERM vibrations and audio feedback accurately
- Users do not have significant auditory or tactile impairments

These assumptions are justified as tilt interaction is common in games and consumer devices, and the system is designed for able-bodied users.

Environmental Assumptions

- The system is used in a quiet indoor environment with minimal background noise
- The user is stationary and positioned in front of a monitor

This ensures that audio feedback is clearly perceivable and that tilt-based control remains precise.

3.1 Usage model

The final version of the game was implemented using a router to connect the PC and the Raspberry Pi. However, during the experimental sessions with participants, the entire system was run locally for convenience. The system functions correctly in both configurations, requiring only minimal modifications, such as changes to the IP address and serial port settings.
When the game starts, a window appears on the screen prompting the player to enter their name. After confirming, the game begins and the labyrinth is displayed on the monitor. The player holds the gamepad with both hands and keeps it in a comfortable, neutral position.
The game can be executed in three different modalities: video only, video + audio, video + audio + haptic feedback.

Basic Interaction

The game is controlled entirely through tilting the gamepad.
The player does not need any technical knowledge of sensors, audio systems, or hardware components.

- Tilt forward / backward → the ball moves forward or backward
- Tilt left / right → the ball moves left or right

The on-screen labyrinth tilts accordingly, causing the ball to roll under gravity. The goal is to navigate the ball through the maze, avoid holes, and reach the goal area.

Feedback and User Interface

The system provides immediate multimodal feedback to help the player understand what is happening in the game:

Visual Feedback

- The monitor displays the labyrinth, the ball, holes, and the goal area in real time.
- The movement of the ball directly reflects the tilt of the gamepad.

Audio Feedback

Sound effects provide intuitive information about the ball's state:

- Rolling sound: becomes louder and faster as the ball's speed increases
- Wall collision: produces a bouncing sound
- Falling into a hole: triggers an explosion sound
- Reaching the goal: plays a victory sound

Vibrotactile Feedback

Vibration is used to communicate danger and collisions:

- Near a hole: vibration intensity increases as the ball gets closer
- Wall collision: vibration accompanies the bounce

This allows the player to perceive critical events even without constantly looking at the screen.

Gameplay Flow

1. Enter player name and attempt number
2. Hold the gamepad and tilt it to control the ball
3. Use audio and vibration feedback to avoid hazards
4. Reach the goal to complete the level

Usability Considerations

The interaction is designed to be simple, intuitive, and immediately understandable.
Because the system relies on natural tilt gestures and perceptual feedback, users can focus on gameplay rather than learning controls.

4 Implementation

The core components of the system have been fully implemented, including tilt-based input, real-time rendering, audio feedback, and vibrotactile feedback.
The gamepad firmware was implemented on a Teensy 3.6 microcontroller using the Arduino framework. Accelerometer data from the ADXL337 is continuously sampled, calibrated at startup, and transmitted to the PC via OSC communication. The same channel is used to receive commands for controlling the ERM vibration motors.
The system follows a distributed architecture: the PC and the Raspberry Pi are connected through a router forming a local network. The user interacts with the system through a physical controller connected to the Raspberry Pi, while the visual interface is displayed on the PC. Audio feedback is reproduced through a sound card and loudspeaker connected to the Raspberry Pi.
On the PC side, the game logic, physics simulation, and rendering were implemented in Python using Pygame for window management and input handling, and PyOpenGL for real-time 3D rendering of the labyrinth and ball. Ball movement is computed by mapping the tilt angles to gravity vectors.
Audio feedback was implemented using Pure Data, with four separate blocks responsible for rolling, collision, explosion, and victory sounds. Communication between the Python application and Pure Data was handled via OSC, using the python-osc library.

From an implementation perspective, the most interesting aspects were:

- the real-time synchronization between visual, audio, and haptic feedback, and
- the mapping of continuous sensor input to both physics simulation and sound parameters.

Simple optimizations were applied, such as sensor value smoothing, event-based audio triggering, and threshold-based vibration updates to reduce unnecessary OSC messages. Relevant code snippets for each subsystem are provided in the Appendix.

5 Evaluation

Experimental Setup and Hypotheses

Hypotheses

- H1: Multimodal feedback reduces task completion time.
- H2: Multimodal feedback reduces the number of errors (falls into holes, collision with walls).
- H3: Multimodal feedback improves perceived immersion and usability.

Experimental Design

Within-subject design, following course guidelines, every subject tests the game with random conditions to ensure to mitigate the learning effect.
Conditions:

- Visual-only
- Visual + Audio
- Visual + Audio + Haptic

Variables

- Independent variable: feedback modality.
- Dependent variables:
    - Completion time.
    - Number of lives lost.
    - Number of wall collisions
    - SUS questionnaire score.
- Control variables:
    - Same maze layout.
    - Same starting position.

Procedure

Each participant has to:

- Read instructions and sign consent.
- Perform a practice session.
- Complete the task in each condition (randomized order).
- Fill the google form questionnaire

Pilot Study

Before running the main experiment, a pilot study was conducted to verify the feasibility of the system and the experimental procedure.
The pilot study involved 2 participants, who tested all three feedback conditions (visual-only, visual + audio, visual + audio + haptic). The main goals of the pilot study were to assess the clarity of the instructions, the stability of the system, and the effectiveness of the multisensory feedback mappings.
Based on the pilot study, several adjustments were made. The intensity of the haptic feedback near the holes was tuned to avoid excessive vibration, while still providing a clear warning signal. In addition, the audio feedback for ball rolling was slightly attenuated to prevent masking other sound events such as collisions and falls. Minor refinements were also applied to the experimental procedure, including a longer familiarization phase to allow participants to better understand the tilt-based control mechanism.
Overall, the pilot study confirmed the viability of both the system and the experimental protocol, and helped refine the interaction parameters before the final evaluation.

Results

Participants and Procedure

Participants were recruited through convenience sampling, including friends, family members, and volunteers who expressed interest in contributing to the study.
Each experimental session lasted approximately 30 minutes per participant and included:

- reading the information sheet,
- signing the consent form,
- a training session to familiarize with the system,
- the experimental task, and
- completion of the final questionnaire.

The information sheet clearly explained how the game works, the objective of the task, and the overall structure of the experimental procedure.
All participants signed an informed consent form prior to the experiment. No participant chose to interrupt or withdraw during the experimental session.

Questionnaire Structure

The post-experiment questionnaire was composed of five sections:

1. Demographic information
2. Usability evaluation using the System Usability Scale (SUS)
3. Evaluation of the hypotheses through questions related to perceived multisensoriality
4. Emotional assessment using the Self-Assessment Manikin (SAM)
5. Open-ended questions

Most questionnaire items used a 5-point Likert scale (from strongly disagree to strongly agree).
For the SAM questionnaire, a 9-point scale was used.

The order of the questions was kept identical for all participants, as it was not expected to influence the responses.
The sections of the questionnaire are described below:

- *Section 1: Demographics*

Participants were aged between 18 and 60 years, with a similar number of male and female participants. Experience with similar technologies or games varied across participants, ranging from low to high.

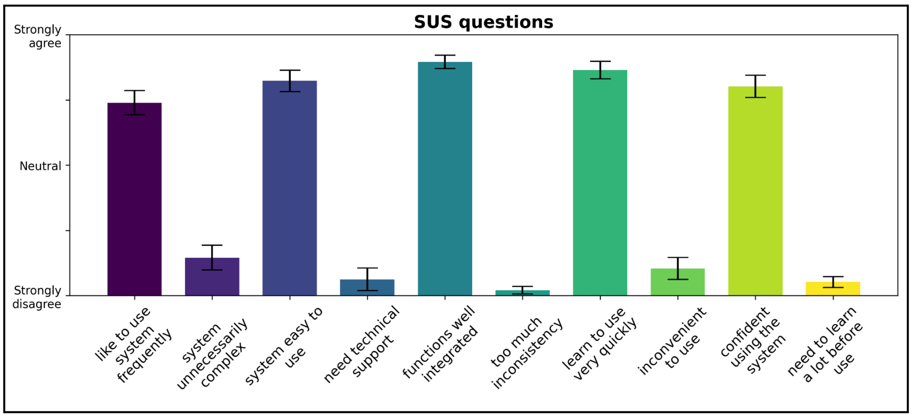
- *Section 2: Usability Evaluation (SUS)*

Usability was evaluated using the 10 standard items of the System Usability Scale (SUS). The SUS score was computed using the standard formula:

$$X = \sum points\ of\ odd\ question - 5 \qquad Y = 25 - \sum points\ of\ even\ question$$

$$SUS\ Score = (X + Y) \cdot 2.5$$

With our results, SUS score is equal to 87.4



This result indicates well above-average usability, suggesting that the system demonstrates excellent perceived usability.

- *Section 3: Hypotheses Evaluation: Multisensoriality*

This section aimed to evaluate perceived multisensoriality in relation to the three initial hypotheses:
H1: Multimodal feedback reduces task completion time.
H2: Multimodal feedback reduces the number of errors (e.g., falling into holes, collisions with walls)
H3: Multimodal feedback improves perceived immersion and usability.

The results indicate that, for H1 and H2, the presence of multimodal feedback contributed to faster task completion and helped reduce errors such as collisions and falls. In particular, H3 is strongly supported by participants' responses, suggesting that multimodal feedback significantly enhances perceived immersion and usability.

In addition to subjective questionnaire data, the hypotheses were also evaluated using objective performance metrics extracted from the experimental logs. These metrics are described in detail in a dedicated section.

- *Section 4: Emotional Evaluation (SAM)*

The Self-Assessment Manikin (SAM) questionnaire was used to evaluate participants' emotional responses while interacting with the system, along the dimensions of valence, arousal, and dominance.

The averaged results we collected are the following:

- Valence: 7.67
- Arousal: 7.54
- Dominance: 7.29

The results show high valence, arousal, and dominance scores, indicating that participants generally experienced positive, highly engaging, and strongly controllable emotional states during the interaction.

- *Section 5: Open-Ended Questions*

At the end of the questionnaire, participants were asked to freely comment on aspects they appreciated, aspects they disliked, and possible improvements to the system. A thematic analysis was conducted to identify recurring patterns across responses.
Overall, participants reported a strong sense of immersion, supporting H3. Many users highlighted the physical and embodied interaction enabled by the gamepad as particularly engaging:

> *"Mi aspettavo di muovere la pallina con il movimento, invece il movimento del piano è molto più coinvolgente." (P3)*
> *"Un gioco nuovo, semplice da usare. Mi è piaciuto che per inclinare il labirinto si usasse qualcosa di 'fisico'." (P5)*

These comments indicate that physical manipulation significantly enhanced engagement and perceived immersion.

Regarding negative aspects, a recurring theme concerned initial usability issues, particularly related to controller sensitivity during the first interaction:

> *"Alcune difficoltà al primo approccio con la sensibilità del controller." (P8)*

A smaller number of participants also mentioned that vibration feedback was perceived as too strong or less helpful during gameplay.

Multisensory feedback was generally evaluated positively; however, some users reported preferring auditory feedback over vibration:

> *"Ti immergono di più nel gioco ma mi sentivo più sicuro senza vibrazioni." (P12)*

Finally, responses to the question "What would you improve?" revealed a clear and consistent desire for more content and higher difficulty, including additional levels, longer paths, and more varied obstacles:

“Aggiungere livelli, aggiungere ostacoli diversi, non solo buchi.” (P1)
“Mi piacerebbero percorsi più lunghi.” (P2)
“Vorrei più livelli, anche più difficili.” (P4)

Rather than indicating dissatisfaction, these suggestions suggest that participants were engaged and motivated to continue playing, reinforcing the positive impact of multimodal feedback on immersion and usability, in line with our hypothesis.

Performance Metrics and Composite Score

In addition to questionnaire data, the prototype automatically recorded participants' performance during the task.
The following data were saved in a CSV file for each trial:

$Name,\ Trial,\ Mode,\ Level\_Reached,\ Outcome,\ Total\_Time\_sec,\ Wall\_Collisions,\ Remaining\_Lives$

To summarize participants' performance into a single metric, a composite performance score was defined by combining completion time, number of collisions, and remaining lives.

The normalized components (time T, collision C, lives L) were computed as follows:

$$Tnorm = \frac{1-(T-Tmin)}{(Tmax-Tmin)} \qquad Cnorm = \frac{1-(C-Cmin)}{(Cmax-Cmin)} \qquad Lnorm = \frac{L}{Lmax}$$

The final score was calculated as:

$$Final\ Score = W_t \cdot Tnrom + W_c \cdot Cnrom + W_l \cdot Lnrom$$

$$\text{where } W_t = 0.4; \quad W_c = 0.3; \quad W_l = 0.3$$

The following table summarizes the results for each experimental condition: video-only, video + audio, and video + audio + haptic feedback:

	Time		Collision		Remaining Lives		Composite Score	
Mode	Mean	Std Dev.	Mean	Std Dev.	Mean	Std Dev.	Mean	Std Dev.
Video	110,07	29,84	68,58	108,36	1,75	1,26	0,61	0,19
Video + Audio	**98,57**	31,21	51,96	85,14	1,75	1,45	**0,66**	0,17
Video + Audio + Haptic	106,84	23,70	**46,58**	48,97	**1,79**	1,18	0,65	0,18

Bold: best result Underlined: second best result

From the results, it can be observed that the addition of audio feedback leads to improvements across all performance metrics (completion time, number of collisions, remaining lives, and consequently the Composite Score). This suggests that the inclusion of auditory feedback positively enhanced overall gameplay performance.

Furthermore, the addition of haptic feedback also resulted in better performance compared to the visual-only condition. However, the improvement was less pronounced, as several participants reported experiencing increased pressure when haptic feedback was present.

Overall, the results indicate that the inclusion of multisensory feedback improves player performance compared to relying on visual feedback alone.
(For the plot of this results, see the Code Appendix)

Based on the collected data, no significant correlations were found between the demographic variables and the multisensory-related questions. Moreover, due to the limited sample size, no inferential statistical tests were performed.

6 Discussion and conclusions

The results emerging from the evaluation suggest that the integration of multimodal feedback has a positive impact on both user performance and subjective experience. In particular, the combination of visual, auditory, and haptic cues appears to improve users' spatial awareness within the maze. Participants were better able to perceive the position of the ball and the surrounding obstacles, especially in critical situations such as proximity to holes. This increased awareness resulted in fewer errors, as users could anticipate dangerous areas earlier and adjust their movements more effectively.

In addition to performance-related improvements, multimodal feedback also contributed to higher levels of user confidence and engagement. Continuous auditory feedback mapped to the ball's movement helped users maintain a sense of control, while vibrotactile cues reinforced the physicality of the interaction. Together, these modalities enhanced immersion and made the interaction feel more natural and responsive compared to a visual-only condition.

Despite these encouraging results, several limitations must be acknowledged. The study involved a small number of participants, which prevents drawing statistically significant conclusions. Furthermore, the sound design, although functional, remained relatively simple and could be expanded to convey richer information or more nuanced emotional feedback.

Several important lessons were learned during the development and evaluation of the system. Calibration and signal smoothing of the accelerometer data proved to be crucial for ensuring a stable and usable interaction, as even small amounts of noise could negatively affect user control. Moreover, continuous feedback, such as rolling sounds and proximity-based vibration, was found to be more informative and effective than purely discrete event-based cues. In particular, haptic feedback was especially beneficial near hazardous areas, as it provided an immediate and intuitive warning without overloading the visual or auditory channels.

Future work could extend the project in several directions. The maze design could be made more complex and adaptive, for example by dynamically adjusting difficulty based on user performance. The sound design could be further developed to include more expressive or context-aware audio cues. A larger-scale user study would be necessary to validate the findings quantitatively and assess statistical significance. Finally, the system could be adapted to application domains beyond entertainment, such as motor skill training or rehabilitation, where multisensory feedback may support learning and recovery processes.

Group members contributions

The project was developed through close collaboration, with team members consistently working in pairs across all phases of the work. Design decisions, implementation, and testing were shared to ensure coherence between software, hardware, and interaction components.

Riccardo Zannoni focused primarily on the software side of the project. His main contributions included the overall software architecture developed using Pygame, sound design, and the creation and integration of Pure Data patches for audio processing and interaction.

Nicola Cappellaro was mainly responsible for the hardware aspects of the project. His contributions included hardware design, 3D modeling and printing of the physical case, and the implementation of OSC communication between the hardware components and Pure Data.

References

Assess the results: slide of the course
Build the questionnaire: slide of the course
Composite Scores / Performance Indices: Url
Design and conduct the experiment: slide of the course
GDPR – Data Protection (for questionnaires): Url
Min–Max Normalization (Feature Scaling): Url
Self-Assessment Manikin (SAM): slide of the course
Sus computation: url
System Usability Scale (SUS): slide of the course

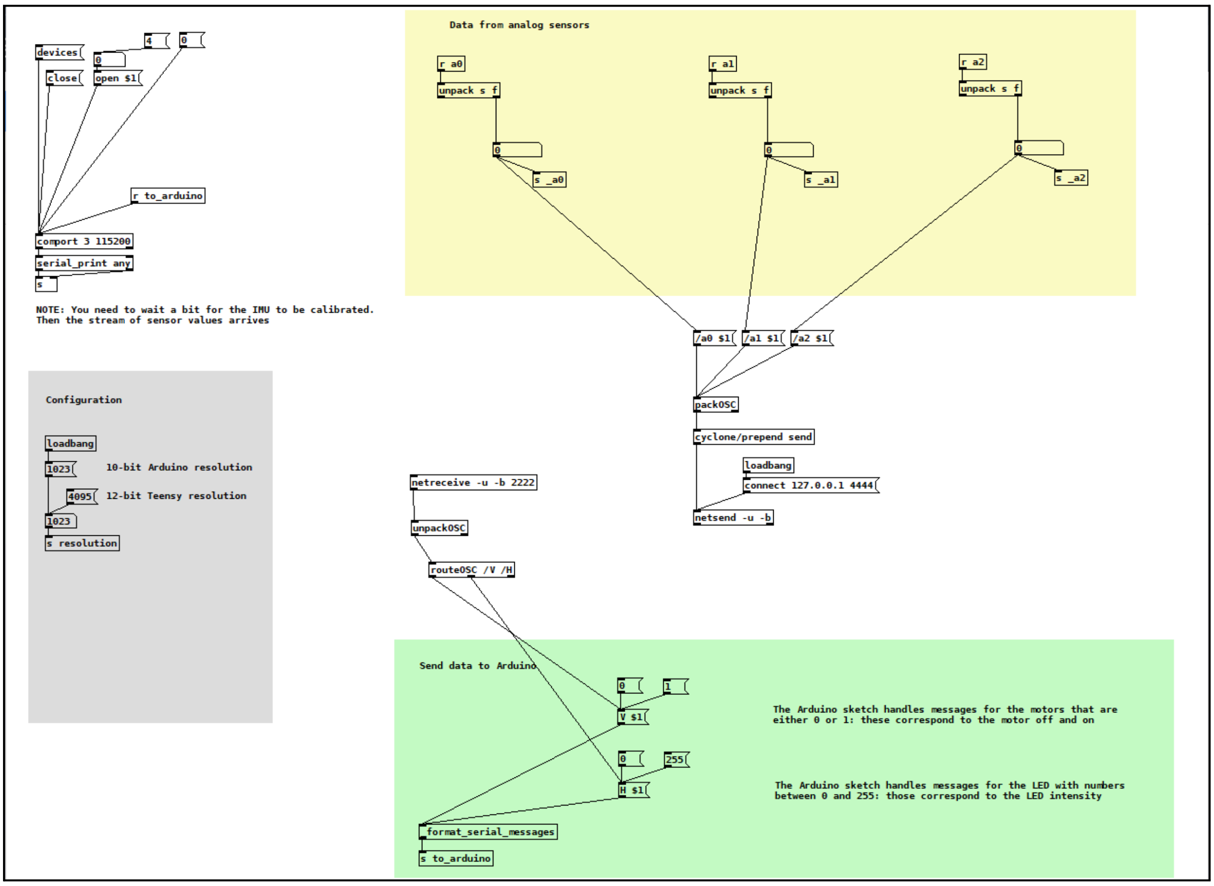
Code appendix

The full source code and all Pure Data patches are available in the online repository at https://github.com/rickyzanna02/MazeTilt.
Below, we include the most relevant portions of the code for clarity.

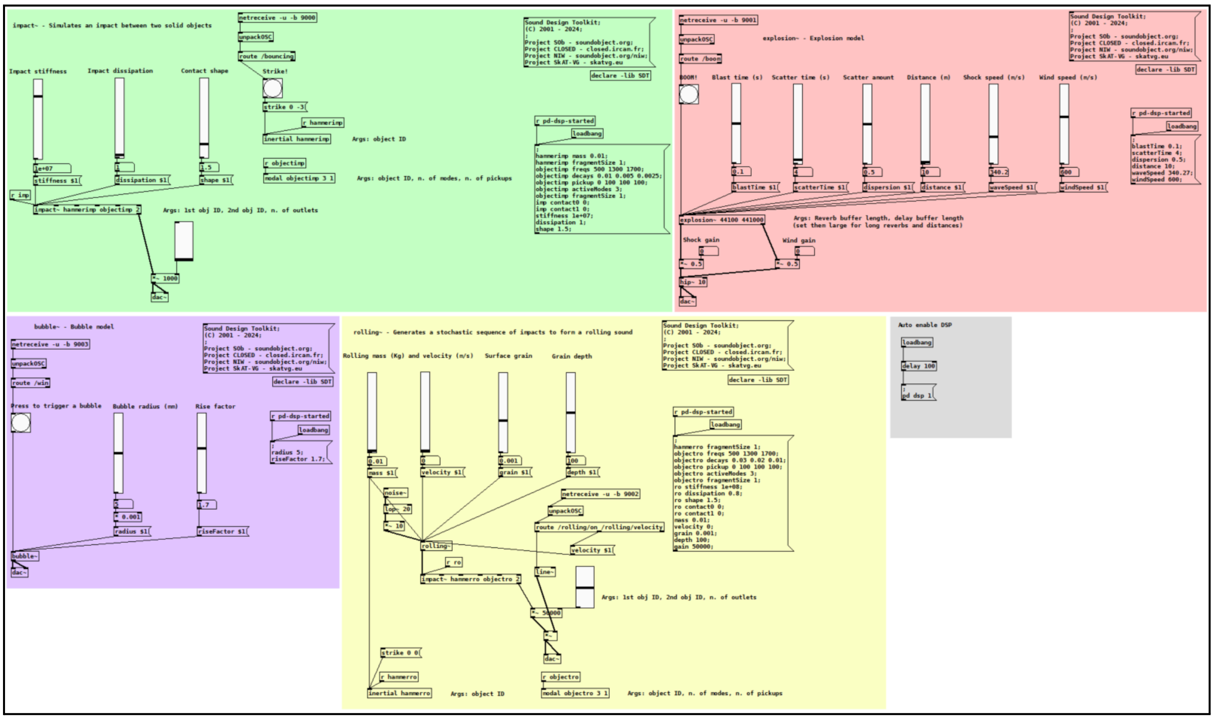
- Accelerometer update function that implements tilting:

```
def update(self):
        xyz = self.read_latest_xyz()
        if xyz is None:
            return (self.tilt_x_deg, self.tilt_z_deg)

        x, y, z = xyz

        # Initial offset calibration
        if not self.calibrated:
            self._sumx += x
            self._sumy += y
            self._sumz += z
            self._calib_count += 1
            if self._calib_count >= self.calib_samples:
                self.ox = self._sumx / self._calib_count
                self.oy = self._sumy / self._calib_count
                self.oz = self._sumz / self._calib_count
                self.calibrated = True
            return (self.tilt_x_deg, self.tilt_z_deg)

        # Remove offset
        ax = x - self.ox
        ay = y - self.oy
        az = z

        roll_deg  = math.degrees(math.atan2(ax, math.sqrt(ay*ay + az*az)))
        pitch_deg = math.degrees(math.atan2(ay, math.sqrt(ax*ax + az*az)))

        target_tilt_x = -pitch_deg
        target_tilt_z = roll_deg

        # Deadzone
        if abs(target_tilt_x) < self.deadzone_deg:
            target_tilt_x = 0.0
        if abs(target_tilt_z) < self.deadzone_deg:
            target_tilt_z = 0.0

        # Smoothing
        a = self.smooth
        self.tilt_x_deg = (1 - a) * self.tilt_x_deg + a * target_tilt_x
        self.tilt_z_deg = (1 - a) * self.tilt_z_deg + a * target_tilt_z

        return (self.tilt_x_deg, self.tilt_z_deg)
```

- Pure Data patch for receiving accelerometer values:



- Pure Data patch for audio feedback (rolling, wall impacts, falling into holes, and winning the level):



- Plot of the metrics results: time, collision, remaining lives, composite score