Robot System Investigations: Manual vs Autonomous Controls

A Study on Human Preferences on Manual or Autonomous Robotic Control Systems

Phi Bui, Ju-Hung Chen, Joey Grossman, Saurya Koka

December 24, 2024

This study explores whether humans prefer manually controlling robots over using autonomous options. Using four Sphero Bolts, participants placed obstacles on a map and guided the robots manually or allowed autonomous navigation. Results supported the hypothesis that humans favor manual control, as it enabled faster task completion, fewer collisions, and greater precision. In contrast, autonomous control was slower and less efficient, reducing participants' trust. These findings suggest that personal control and adaptability play a significant role in user preference.

1 Introduction

Robots are increasingly taking on tasks traditionally performed by humans. In many cases, they are not only matching human capabilities but often surpassing them in efficiency and precision. However, humans sometimes prefer to perform tasks themselves since human could have greater control to match personal satisfaction. Take self-driving cars for example, while they provide correction to human errors and increase safety, some people still prefer to drive by themselves because it provides the ability to make intuitive decisions in complex situations. Thus, the goal of this study is to explore whether people still prefer to manually control the robot, even when autonomous options are available.

To address the stated problem and identify people's preference on controlling robots, our group used four Sphero[1] Bolts as the robotics platform. People will be asked to randomly place obstacles on the map, then guide the robot through them manually, and finally watch the robot to navigate autonomously to the goal.

[1]A Sphero is a type of spherical robot designed for K12 education.

This study hypothesizes that humans prefer controlling the robot themselves. Real-world tests with humans showed results that align with this hypothesis. The preference was likely effected by the significantly faster and more efficient performance observed under manual control, as participants could adapt quickly to obstacles and navigate the map with greater precision. In contrast, the autonomous control system exhibited slower responses and struggled to complete the task efficiently, leading to longer completion times and reduced participants' trust in its capabilities.

2 Background

This research examines human preferences for manual versus autonomous control of robots within the context of swarm robotics and human-robot interaction (HRI). By integrating probabilistic modeling, SLAM, and swarm coordination, the study aims to contribute to the growing body of work addressing the challenges of human-robot collaboration.

SLAM and probabilistic modeling form the technical foundation of this project. Hu [4] explored SLAM implementations with a single Sphero robot, focusing on trajectory-based corner detection and mapping. While effective, the approach was constrained by the Sphero's limited sensors and lack of scalability. Hu hypothesized that combining multiple robots and better sensors could enhance SLAM performance. This project builds on this idea by utilizing Monte Carlo Localization (MCL) and Sphero Bolt swarms to address these limitations.

Probabilistic modeling also extends to swarm robotics. Hu et al. [5] introduced SwarmPRM, a novel motion planning algorithm that leveraging probabilistic road-maps and conditional value-at-risk collision avoidance. Their approach demonstrated scalability and robustness under uncertain conditions, which informs our use of probabilistic road-maps to coordinate Sphero robots. Similarly, Kegeleirs et al. [6] developed Mercator, an enhanced SpheroRVR platform, and highlighted the potential of external localization systems like cameras for swarm coordination. This aligns with our use of overhead vision for tracking and controlling Sphero Bolt robots. Wustrau [9] extended these ideas by applying swarm intelligence principles to homogeneous robot platforms, showing how local interactions among robots could yield emergent behaviors.

Human-robot interaction introduces additional complexity, as human behavior is often unpredictable. Knox et al. [7] demonstrated how collecting initial user data can improve robot communication, emphasizing the importance of building an understandable model of the user. Gui et al. [2] explored probabilistic movement modeling to allow for better human-robot interaction, using Gaussian processes to infer user behavior. These approaches inform our system design, where participants interact with robots in both manual and autonomous modes, requiring real-time adaptability.

Trust in robotic systems is also a critical factor in HRI. Knox et al. [8] examined trust-aware motion planning, showing how probabilistic models of human trust can enhance robot collaboration. The role of intuitive robot design in fostering effective HRI is underscored by the findings of Araujo et al. [1], who investigated how interface design impacts user interaction with robotic systems. Their work highlights the importance of clear and accessible control systems, which is a key consideration in our experiment. Trust and visual feedback were integral parts of our study, where we aimed to build an understanding of the user's trust towards robots beforehand via a preliminary survey. Furthermore, we implemented a display to show the user the "robot's point of view" by visually displaying all of the systems we implemented for localization, planning, and calibration.

Finally, NavFormer [3] presented a transformer-based navigation approach, focusing on exploration and obstacle avoidance in dynamic environments. Although unrelated to HRI, it provides insights into state-of-the-art navigation techniques, which inform potential future directions for this project. This is complemented by research such as Araujo et al. [1], who utilized probabilistic multi-modal planning to model human behavior, demonstrating the viability of integrating human predictions into robotic decision-making.

Overall, the current state of research in SLAM, swarm robotics, and HRI emphasizes the importance of adaptability, probabilistic modeling, and user-focused design. By combining these principles, this project examines human preferences for manual versus autonomous robot control. The integration of Monte Carlo Localization, probabilistic road-maps, and visual feedback systems aims to bridge the gap between technical performance and user satisfaction, contributing to a deeper understanding of human-robot collaboration.

3 Methodology

To assess human preferences between manual and autonomous robot control systems, an experiment was conducted using a scenario where participants guided two Sphero Bolt robots through a human-designed obstacle course. The course was created by the participants themselves, featuring hand-placed orange sticky notes as obstacles and a goal marked with purple tape, as illustrated in Figure 1.

Figure 1: Subject initializing obstacles.

3.1 Hardware

This study utilized the Sphero Bolt, a programmable robot developed by Sphero Inc. To facilitate quick switching between manual and autonomous control modes, four Sphero Bolts were employed. Two of the robots were connected to separate mobile phones that run simple programs via the Sphero Edu app, while the other two were linked to a Python-based server.

A top-down view of the navigation environment was captured using an Akaso Brave 4 camera, which recorded 4K images (2880×2160 resolution) and transmitted them to the server. The server infrastructure, consisting of three separate servers (detailed in Section 3.3), was hosted on an Asus Zenbook 13 UX325EA-DH51 laptop.

The navigation environment was simulated using a large sheet of paper, marked with colored tape to represent various features and obstacles.

3.2 Libraries

The implementation of the system relied on several Python and JavaScript libraries, each serving a critical role in various aspects of the project. The following libraries were utilized:

- **opencv-python:** Used for image processing and computer vision tasks, including camera-based localization and mapping.
- **websockets:** Enabled real-time communication between the WebSocket server and other components of the system.
- **scikit-learn:** Provided tools for data analysis and algorithm development.

- **numpy:** Used for numerical computations and array manipulation.
- **scipy:** Supplemented `numpy` with advanced scientific computing capabilities, such as optimization and signal processing.
- **spherov2:** Facilitated interaction and control of Sphero robots.
- **pillow:** Used for image processing, including visualization and display functionalities.
- **bleak:** Supported Bluetooth Low Energy (BLE) communication, necessary for interfacing with Sphero robots.

These libraries collectively enabled the development of a robust, efficient, and modular system, providing essential tools for tasks ranging from robot control to data processing.

3.3 Server Architecture

The system architecture consists of three micro-service servers, with the data flow outlined in Figure 2. Each server has a specific role, as detailed below:

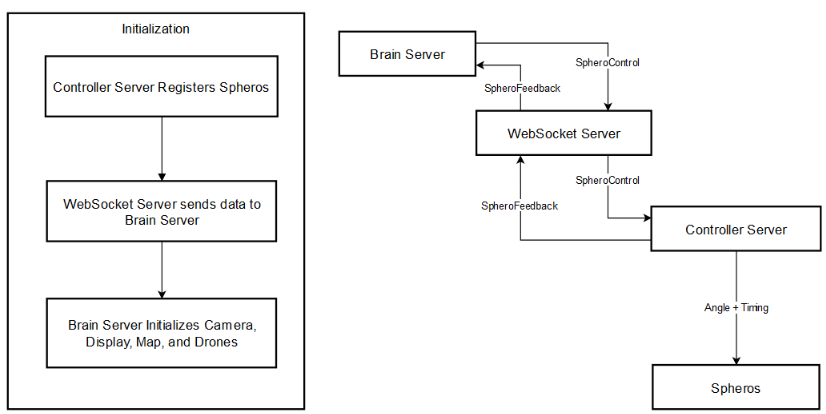


Figure 2: Server architecture data flow.

- **WebSocket Server:** Implemented in JavaScript, this server acts as a centralized communication hub. The server facilitates bidirectional data exchange between the control server and the brain server, ensuring seamless communication across the system.
- **Control Server:** Implemented in Python, the control server is responsible for managing the Sphero robots. Its main functions include:
  - Initializing by sending a list of Spheros and their associated colors to the WebSocket server.

  - Receiving direction and angle commands for each Sphero from the WebSocket server and executing them.
  - Sending feedback messages to the WebSocket server to confirm the successful completion of actions.

  To handle multiple robots efficiently, the control server runs a separate process for each Sphero. Communication between these processes is managed using a message queue, enabling parallel control and scalability.

- **Brain Server:** Implemented in Python, the brain server manages high-level operations, including:
  - Initializing a camera and display process.
  - Managing a map system and localization for each Sphero, encapsulated in a `Drone` class.
  - Utilizing the camera for data acquisition, which is processed by the Map and Localizer.
  - Displaying processed data via the display process.

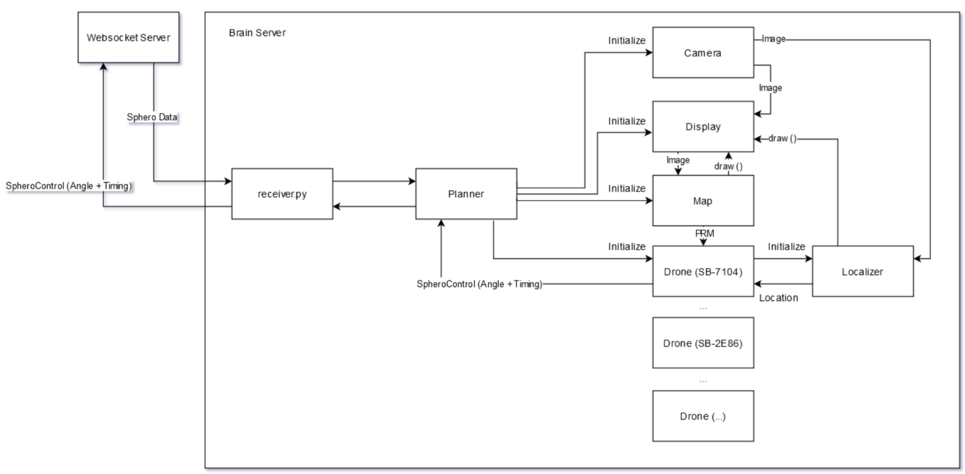


Figure 3: Brain server data flow.

The data flow specific to the brain server is shown in Figure 3. Detailed descriptions of the mapping and localization algorithms are provided in Section 4.

3.4 Study Design

Hypothesis: Humans prefer manual control over autonomous control.

- Pre-experiment questions:

  - Basic information: age, field of major
  - Do you have experience of interacting with robots or controlling the robot?
  - On a scale of 1-10, How much are you willing to trust robots? 0: I am not willing to trust them, 10: I am willing to trust them
  - On a scale of 1-10, How much do you trust robots? 0: I do not trust them, 10: I trust them.
  - On a scale of 1-10, How much would you trust a self driving cars to transport you to your required destination? 0: I do not trust them, 10: I trust them.

- Experiment:
  - Environment setup: Our study was planned to use an open space, Nolop Maker Space, located at Tufts University's Tsungming Tu Complex (Science and Engineering Complex). The camera was mounted on a wooden dowel and secured on a bench for the top down view on the map. Lights were dimmed for better performance on Sphero's LED matrix. The participant was asked to stand in front of the map.
  - Procedure:
    - Ask test subject to fill out pre-experiment questions
    - Explain the task (reach to goal) and the meaning of Sphero's light to the test subject
    - Place 2 Spheros on the start point of the map
    - Ask test subject to place the orange obstacles (Post-it note) on the map (as seen in Figure 1)
    - Ask test subject to use the stick to manually guide the Sphero to the goal (as seen in Figure 4)
    - Record the time of Spheros reach to goal
    - Reset the map but keep the orange obstacle (Post-it note) on the map
    - Run Spheros autonomously and record the time of getting from start to goal
    - Record the time of Spheros reach to goal
    - End the test and ask test subject to fill out post experiment questions

- Post-experiment questions:
  - Please input Sphero's time of completing the course
  - Please input how may orange block the Spheros hit.
  - Which way do you think is easier and makes you feel you have more control? Autonomous or manual?

– On a scale of 1-10, How good do you think Sphero's ability guiding themselves to the goal? 0-Very bad, 10-Very good

– On a scale of 1-10, Do you think Spheros are easy to guide manually? 0-Not easy, 10-Straight forward

– Do you have any trouble controlling Sphero's direction?

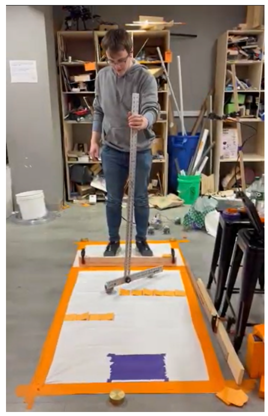


Figure 4: Subject guiding Spheros.

Demonstration Video

A video demonstration of the experiment showcasing both manual and autonomous control modes, along with participant interactions, is available on YouTube (https://youtu.be/9G3ZlSHARaE). The video provides a clear illustration of the experimental setup, navigation performance, and the visual feedback subjects saw during testing.

4 Algorithm

4.1 Algorithm Overview

The proposed algorithm facilitates coordinated localization, path planning, and movement for a swarm of Sphero robots in dynamic environments. It integrates **Monte Carlo Localization (MCL)** for precise position estimation, **Probabilistic Road-map (PRM)** for efficient path planning, and **custom calibration algorithm** for feedback-based controls.

4.2 Swarm Localization and Path Planning Algorithm

Inputs:

- R: Set of Sphero robots

- Probabilistic Road-map (PRM)
- Sensor Data (Images from Camera, Angles from Onboard Compass)

Outputs:

- Angle and Timing necessary to reach a target point for a given robot

Algorithm 1 High-Level Movement Parameter Calculation Algorithm

Initialize:
    Set initial parameters:
      last_location ← None,
      speed ← 0,
      angle_offset ← None.
    Target position from next node along the PRM.
Position Estimation:
Current position $(\text{current\_y}, \text{current\_x})$ is estimated via Monte Carlo Localization (MCL) with confidence.
Check for First Move:
if this is the first move (i.e., last_location = None) **then**
    Set last_location ← $(\text{current\_y}, \text{current\_x})$.
    Return initial angle and timing.
end if
Calculate Movement Vector:
Compute actual movement vector $\Delta x_{\text{actual}}, \Delta y_{\text{actual}}$ from previous to current position.
Compute the actual movement angle actual_angle and distance moved.
Initialize Angle Offset (if needed):
if angle_offset = None **then**
    Set angle_offset ← actual_angle.
end if
Update Speed:
Adjust movement speed using confidence-weighted interpolation:

$$\text{speed} \leftarrow (1 - \text{confidence}) \cdot \text{speed} + \text{confidence} \cdot \frac{\text{distance\_moved}}{\text{timing}}$$

Calculate Target Angle and Timing:
Compute movement vector to the target position $(\Delta x_{\text{target}}, \Delta y_{\text{target}})$.
Calculate target angle relative to offset.
Compute corrected timing based on distance to target and current speed:

$$\text{corrected\_timing} \leftarrow \frac{\text{distance\_to\_target}}{\text{speed} + 10^{-6}}$$

State Update:
Update angle, timing, and last location to current position.
Return angle and timing required to reach target position.
End Algorithm.

4.3 Algorithm Steps

The high-level movement parameter calculation algorithm is divided into the following steps:

1. **Initialization:**
    - Set initial parameters:
        - `last_location` is set to `None`.
        - `speed` is initialized to 0.
        - `angle_offset` is set to `None`.
    - Identify the target position as the next node along the PRM.
2. **Position Estimation:**
    - Use the Camera and MCL to estimate the robot's current position $(\text{current\_x}, \text{current\_y})$ with a confidence value.
3. **Handling the First Move:**
    - If this is the first move (`last_location` is `None`), set `last_location` to the estimated current position.
    - Calculate and return the initial movement angle and timing for the first move.
4. **Movement Vector Calculation:**
    - Compute the actual movement vector $(\Delta x_{\text{actual}}, \Delta y_{\text{actual}})$ between the previous and current positions.
    - Derive the actual movement angle and the distance moved.
5. **Angle Offset Initialization:**
    - If `angle_offset` is still `None`, set it to the computed `actual_angle`, which is the difference between the robot's onboard Compass measurement of 0°and the coordinate system's 0°.
6. **Speed Adjustment:**
    - Update the robot's speed using confidence-weighted interpolation:

    $$\text{speed} \leftarrow (1 - \text{confidence}) \cdot \text{speed} + \text{confidence} \cdot \frac{\text{distance\_moved}}{\text{timing}}$$

    - Confidence level of the current position measurement is derived during the MCL process.
7. **Target Angle and Timing Calculation:**
    - Compute the movement vector $(\Delta x_{\text{target}}, \Delta y_{\text{target}})$ to the target position.
    - Calculate the target angle relative to the angle offset.

- Determine the corrected timing based on the distance to the target and the current speed:

$$\text{corrected\_timing} \leftarrow \frac{\text{distance\_to\_target}}{\text{speed} + 10^{-6}}$$

8. **State Update:**
   - Update the robot's angle, timing, and `last_location` to reflect the current position.
   - Return the corrected angle and timing for the next movement.

4.4 Integration with the System

The algorithm's modular structure allows seamless integration with the swarm management system. Localization, path planning, and command execution processes operate in parallel, ensuring robust and efficient navigation across the swarm.

4.5 Communication and Feedback

The WebSocket server acts as a centralized hub for command and feedback exchange between the control server and the Sphero robots. Each Drone object (representing an individual robot) utilizes algorithm described above to submits a trajectory to the centralized Planner class. The Planner determines if the path is valid (e.g. contains no robot collisions) and makes adjustments as needed. If all paths are valid, the Planner sends the path to the robots via the WebSocket server. The Sphero robots send back feedback confirming they have completed their last action via the WebSocket server, which informs the Planner to start the next move, repeating the process.

4.6 Figures and Visualization

Figure 5 illustrates the MCL process, with the larger circles representing a Gaussian Mixture Model of where the system thinks the robot is. Figure 6 shows an example PRM-generated path. These figures highlight the key components of the proposed algorithm, including localization accuracy and path planning efficiency.

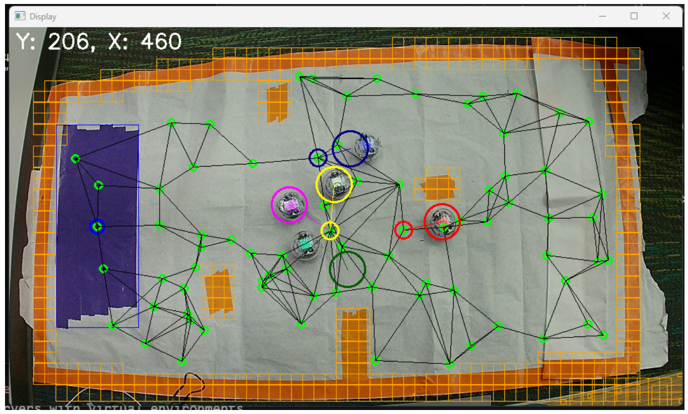


Figure 5: Three Spheros being located.

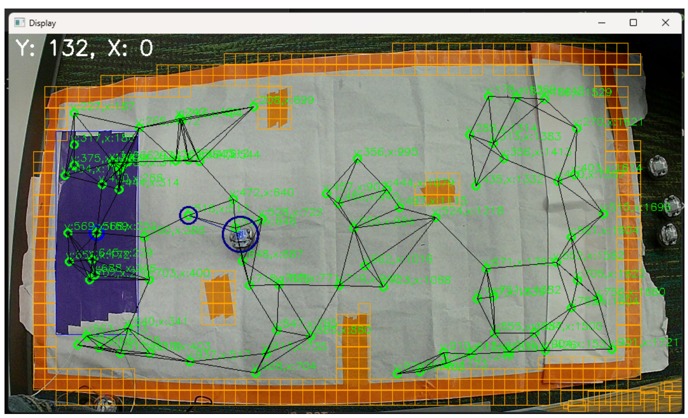


Figure 6: Probabilistic Road-map Generation.

5 Results

5.1 General Background of Participants

Our group collected test data from 11 test subjects, age ranging from 18 to 28. Most of the test subject has engineering related major but half of them have little to none experience of controlling robots. The general background of the participants is shown as Figure 7.

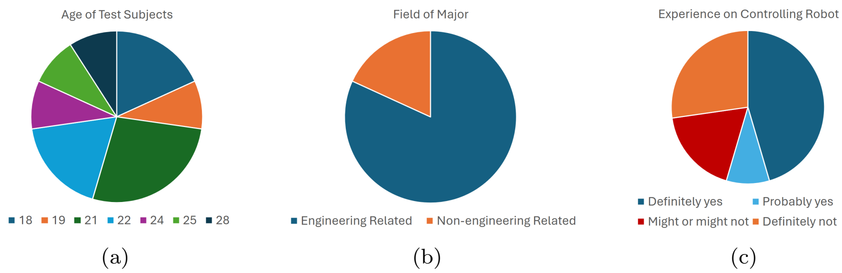
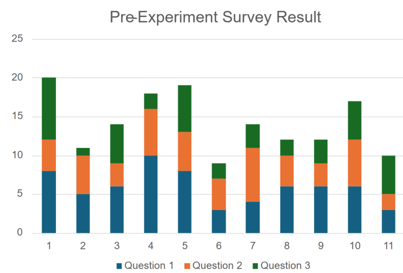


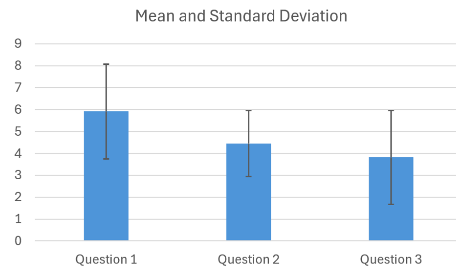
Figure 7: General Background of Participants

5.2 Pre-experiment Survey

As shown in Figure 8, the result of pre-experiment Qualtrics survey shows that participants have mixed feeling of autonomous robots. For Question 1, asking the participant's willingness to trust robots. The responses varied, with some participants showing high willingness (values close to 10) and others being less willing. On average, Question 1 has the highest scores among all three questions, showing that participants are relatively open to trusting robots. For Question 2, asking how much participants trust robots, generally lower compared to participants' willingness to trust robots. The result points out a gap between participants' openness to trust and their actual trust in current robotic systems. Last the question 3, asking participant's trust in self-driving cars received the lowest scores among the three questions. The responses indicate significant skepticism or hesitation when it comes to trusting autonomous vehicles for personal transport. Overall, Question 1 has the highest mean, with a large standard deviation, indicating mixed responses (some highly willing, some hesitant to trust robots). Question 2 shows a slightly lower mean with moderate variability, reflecting a cautious level of trust. Question 3 has the lowest mean, but with considerable variability, indicating that participants are divided on trusting self-driving cars.



(a)

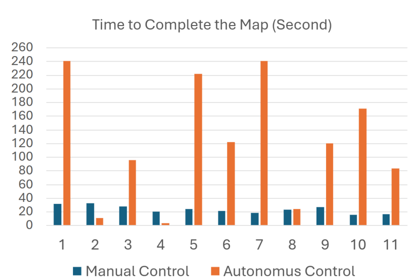


(b)

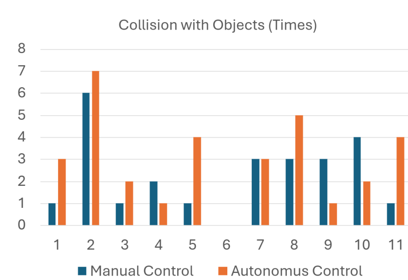
Figure 8: Pre-experiment Survey Result

5.3 Experiment Results

The result of manual control shows no significant difference among all participants. As the result shown in Figure 9a, the time taken for manual control remains consistently low, generally below 40 seconds, with only minor variations. On the other hand, the time taken under autonomous control shows significant variation, with much higher values overall. Several trials (1, 5, 7, and 10) took considerably longer, exceeding 200 seconds, while a few trials (e.g., 2, 4, 8) performed slightly better but still lagged behind manual control. According to Figure 9b, manual control had slightly fewer collisions overall, but there were instances where autonomous control performed better (e.g., trial 10). However, in critical scenarios like trial 2, autonomous control experienced the highest number of collisions (7), compared to 6 for manual control. In the end of experiment, participants rated their experience controlling the robot on a scale of 10. Manual control received an average score of 4.5, while autonomous control scored noticeably lower at 3.8. This result further proved that the participants' preference for manually controlling the robot.

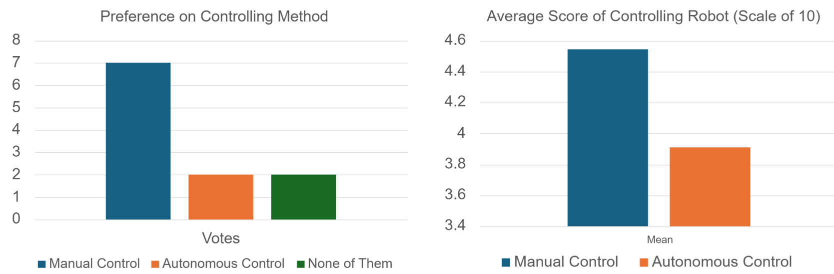


(a)



(b)

Figure 9: Experiment Result



(a) (b)

Figure 10: Experiment Result

Statistical Analysis

To assess the difference in collisions between manual and autonomous control modes, a paired t-test was conducted. Missing values in the dataset were treated as zero collisions, ensuring all data points were included.

The results of the t-test revealed no statistically significant difference in the number of collisions between the two modes ($t(6)$ = -1.146, p = 0.296). This suggests that while participants had a clear preference for manual control, the performance in terms of collision avoidance was comparable between manual and autonomous modes.

This result highlights that factors beyond collision performance, such as user experience and perceived control, likely influenced participant preferences. Further exploration of these factors could provide deeper insights into designing more user-friendly autonomous systems.

6 Discussion

The results of this study reveal an interesting dichotomy between participants' preferences and the perceived intuitiveness of controlling Sphero robots. While the autonomous Sphero navigation system demonstrated similar collision avoidance performance to manual control, participants overwhelmingly preferred manually guiding the robots. This preference persisted despite challenges in intuitively controlling the robots manually.

6.1 Preference for Manual Control

When asked which mode they found easier and offered a greater sense of control, 64% of participants (7 out of 11) indicated a preference for manually controlling the Spheros, while only 18% (2 participants) preferred the autonomous mode. An additional 18% (2 participants) felt that neither mode offered an intuitive experience. This clear preference for manual control aligns with the hypothesis that humans often value a sense of agency, even in situations where autonomous systems perform comparably in collision avoidance.

6.2 Challenges in Manual Control

Despite the preference for manual control, participants generally found guiding the Spheros manually to be non-intuitive. When asked to rate the intuitiveness of manual control on a scale from 0 (not intuitive) to 10 (very intuitive), responses skewed toward the lower end. Notably, 45% of participants rated the intuitiveness as 3 or below, with only 18% rating it at 7. This suggests a significant gap between preference and perceived ease of use.

Participants highlighted several issues with manual control in open-ended responses. Common challenges included:

- Difficulty determining how much force to apply when moving the Spheros (e.g., "I wasn't aware how hard to push them for them to move").
- The Spheros pushing against the manual control tool or rolling beyond the intended distance after a push.
- Increased difficulty in complex courses, with one participant stating, "Manually guiding it with the method presented becomes very high/nigh impossible, especially given that I have to control both of them."

6.3 Trust in Autonomous Navigation

When evaluating the autonomous Sphero navigation, participants provided mixed feedback on its ability to guide itself to the goal. Ratings on a scale from 0 (very bad) to 5 (very good) were distributed evenly, with 54% of responses clustered at 2 or below, indicating dissatisfaction with the navigation system's performance. However, a subset of participants (18%) rated the system as a 10, demonstrating that while some appreciated its capabilities, there remains room for improvement in consistency and reliability.

Participants' low trust in autonomous navigation aligns with broader trends observed in autonomous vehicle research. Despite similar performance, manual control provides a sense of agency that enhances user satisfaction. Incorporating system transparency, such as displaying real-time navigation decisions or providing feedback on path-finding logic, could improve trust and facilitate broader acceptance of autonomous systems.

6.4 Broader Implications

Trust also emerged as a key factor in participants' willingness to rely on autonomous systems. For example:

- When asked about their general trust in robots, ratings varied, with only 20% of participants rating their trust above a 5 on a scale from 0 to 10.

- Trust in self-driving cars as a proxy for autonomous navigation was similarly low, with 55% of participants rating their trust as 3 or below on a 10-point scale.

These findings suggest that even with comparable performance in collision avoidance, the lack of trust and perceived reliability in autonomous systems significantly affects user preference. Improvements in equipment and algorithm precision could narrow the performance gap, particularly in terms of timing, which was identified as an area of improvement.

While this study was conducted in a controlled environment, such as the Nolop Maker Space, it reflects real-world scenarios where robots must navigate dynamic obstacles placed by humans. Future work will explore testing the system in outdoor settings, such as disaster response simulations or urban navigation, to evaluate its performance in more complex, unpredictable environments.

6.5 Limitations and Future Work

While the results provide valuable insights, there are limitations to consider. First, the equipment used in this study, particularly for manual control, introduced challenges that may have influenced participants' experiences. As one participant suggested, "A shallower blade angle like a hockey stick would make handling easier." Future iterations of this study could incorporate more ergonomic tools to enhance manual control.

Additionally, improving the autonomous navigation algorithms and the equipment running it could make the autonomous mode more competitive, particularly in complex obstacle courses. These improvements could also address the timing discrepancies between manual and autonomous modes, further highlighting the strengths of autonomous navigation.

6.6 Conclusion

This study demonstrates that while manual and autonomous performance was similar, participants preferred manually controlling the Spheros. This is further exasperated by their low trust in autonomous systems. Addressing the identified challenges in both manual and autonomous modes could lead to a more balanced evaluation of these control methods. This study is directly applicable to real-world scenarios such as autonomous vehicles and emergency response robotics, where user trust and the ability to manually intervene play critical roles. Understanding human preferences for manual versus autonomous control enables the development of adaptable systems that balance automation with human oversight, improving performance and acceptance in dynamic, high-stakes environments like disaster zones or unfamiliar terrain. Ultimately, this research highlights the importance of user experience and trust in shaping preferences for human-robot interaction, even in scenarios where autonomous systems perform comparably to manual control.

7 Acknowledgments

- **Phi Bui** The authors would like to acknowledge Phi Bui for their significant contributions to this research. Phi was responsible for writing the background, code section of the methodology, algorithm section, discussion section, and bibliography. They also designed the system, procured all the necessary equipment, and developed all of the code required for the system's implementation. Their dedication and effort were instrumental in bringing this project to fruition.
- **Ju-Hung Chen** The authors would like to acknowledge Ju-hung Chen for their contribution to this project. Ju-Hung was dedicated on designing the study of this research, creating survey, organizing data, creating charts, writing introduction, methodology, result. They also programmed LED matrix animation, seeking potential solution to connect to multiple Spheros and setup environment for testing and collecting data. Their commitment and hard work were pivotal in making this project a success.
- **Joey Grossman** The authors would like to thank for Joey Grossman for his attempts to implement Sphero movement.
- **Saurya Koka** The authors would like to thank Saurya Koka for helping set up and take down the experiments.

8 Bibliography

References

[1] S. Araujo and D. Lee. Human-robot interaction: The influence of interface design on user preferences for control modes. *Cognition, Technology & Work*, 22(1):123–134, 2020.

[2] L. Gui and H. Zhou. Probabilistic movement modeling for intention inference in human–robot interaction. *Robotics and Autonomous Systems*, 147:102765, 2024.

[3] Ramin Hasani, John Smith, and Taeho Kim. Navformer: A transformer architecture for robot target-driven navigation in unknown and dynamic environments. *arXiv preprint arXiv:2402.06838*, 2024.

[4] Xiaoxuan Hu. Slam with the sphero robot. Master's thesis, Institut de Robòtica i Informàtica Industrial, Universitat Politècnica de Catalunya, 2018.

[5] Yunze Hu, Xuru Yang, Kangjie Zhou, Qinghang Liu, Kang Ding, Han Gao, Pingping Zhu, and Chang Liu. Swarmprm: Probabilistic roadmap motion planning for large-scale swarm robotic systems. *arXiv preprint arXiv:2402.16699*, 2024.

[6] Mathieu Kegeleirs, Aaron Todd, Sam Garcia, Laetitia Lefèvre, Benoît Garbinato, and Francesco Mondada. Mercator: A versatile and accessible platform for swarm robotics. *Technical Report*, 2022.

[7] T. Knox and G. Harris. Using exploratory search to learn representations for human preferences in robot communication. *Journal of Human-Robot Interaction*, 16(1):35–48, 2024.

[8] T. Knox and X. Wang. Trust-aware motion planning for human-robot collaboration under distribution temporal logic specifications. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1505–1512, 2024.

[9] Christian Wustrau, Sanaz Mostaghim, and Christoph Steup. Building a scalable swarm application using sphero robots. 2020.