Master Internship

Laying the Groundwork for Guiding Children Using EDucational MOdular Robots (EDMO)

Laurent Chen

Internship report submitted in partial fulfillment
of the requirements for the degree of
Master of Science of Artificial Intelligence
at the Department of Advanced Computing Sciences
of the Maastricht University

Internship supervisors:

Rico Möckel

Maastricht University
Faculty of Science and Engineering
Department of Advanced Computing Sciences

October 14, 2025

Abstract

This research investigates the use of EDucational MOdular Robots (EDMO) to engage children in interactive, hands-on learning experiences. The study focuses on developing an automated system to guide children in optimizing the movement of modular robots, specifically in a snake-like configuration. By leveraging GoPro cameras and ArUco markers for motion tracking, the research explores the fitness landscape of EDMOs to identify optimal gait parameters. The application of Powell's optimization method reveals three distinct optimal gaits: "snake-like," "caterpillar-like," and "walk-like." However, challenges in transferring these parameters between different EDMOs due to motor calibration discrepancies and servo performance issues are identified. The study also examines the impact of external factors, such as cables, on the robots' performance and discusses potential solutions, including wireless communication systems. The findings suggest that an interactive system using fitness landscape data can provide personalized feedback to help children achieve optimal gaits, enhancing their educational experience. Future work should focus on refining tracking systems, improving motor calibration, and exploring wireless setups to further enhance the reliability and functionality of EDMOs in educational settings.

Contents

1 Introduction

Educational robotics is an increasingly influential field that uses interactive, hands-on learning to engage children and foster skill development. By creating dynamic and experiential learning environments, educational robotics enhances essential skills such as problem-solving, creativity, and critical thinking [5, 15, 19, 24, 34]. Research has demonstrated that robot-assisted learning not only boosts academic performance but also provides personalized learning experiences, adapting to each student's pace, interests, and cognitive abilities [35]. Programs like LEGO-based robotics have proven to increase motivation, collaboration, and technological self-efficacy [6].

In addition to strengthening STEM (Science, Technology, Engineering, and Mathematics) skills by facilitating the understanding of difficult abstract concepts through physical, manipulable artifacts [7], educational robotics nurtures broader competencies like critical thinking, teamwork, and creativity [12]. These abilities are not only valuable for academic success but are also highly sought-after in professional domains like engineering and computer science [3, 20, 29]. While simply interacting with robots may not always foster technical skills, structured educational robotics programs that involve designing, building, and programming robots have been shown to provide students with practical experience in computational thinking, coding, mechanical systems, and control logic [7, 30].

Recent advances in artificial intelligence have further enhanced the capabilities of educational robots. Intelligent Tutoring Robots (ITRs), for example, leverage AI to perceive students' emotional and cognitive states, plan pedagogical strategies, and adapt their actions to optimize learning outcomes [35]. These robots integrate perception–planning–action loops, enabling real-time interaction with learners and facilitating personalized instruction that goes beyond pre-programmed behaviors. This capacity to model learners and adjust dynamically distinguishes ITRs from more traditional educational tools [1].

Beyond STEM learning, educational robots—including social robots—play a growing role in supporting language acquisition, special education, and social-emotional development [17]. Social robots are anthropomorphic, emotionally expressive agents that engage learners in rich, naturalistic interactions, promoting verbal communication, empathy, and inclusion, especially in children with autism or learning differences [8]. Their embodied presence and ability to engage in multimodal communication have shown to foster meaningful relationships with students, improving both affective and cognitive learning outcomes [17].

As interest in educational robotics grows and more pilot programs are introduced [2], its potential influence spans cognitive, social, and emotional domains of development. Beyond supporting subject mastery, educational robotics can also inspire more children to consider careers in engineering and science [14, 30].

This research investigates EDucational MOdular Robots (EDMO), a modular robotic platform developed at Maastricht University, which exemplifies the potential of educational robotics. EDMO is a flexible, custom-built system designed to support robotics education across various levels. The platform enables students to experiment with a range of robotic configurations, including snake-like, bipedal, and quadrupedal forms (Figure 1.1), encouraging creativity and innovation. This versatility allows students to explore different robotic designs and functionalities based on their interests and challenges.

For advanced students, EDMO introduces complex technical concepts such as kinematics, control algorithms, and sensor integration. For younger learners, it promotes collaboration, communication, and an interest in robotics through teamwork-based activities. By fostering both technical and social skills, EDMO helps prepare students for a technology-driven future.

2 Objective and approach

This research is part of a psychology study designed to explore how children interact and collaborate during tasks that require cooperative effort. In the study, each child is responsible for controlling one of the four legs of an EDMO robot in spider configuration (Figure 2.1a), using an individual interface (Figure 2.1b). The tasks include activities such as "Follow the trail" (on the mat, Figure 4.1b) and achieving a fast gait. Instructions and feedback are provided by the teacher either verbally or through their own interface [28]. However, the current setup demands continuous teacher supervision for each group, limiting the system to either a single active robot at a time or requiring a dedicated teacher per group—both of which are highly restrictive.

The goal of this research is to develop an automated system capable of providing real-time feedback, thereby minimizing the need for constant teacher supervision. The objective is to create a tool that guides

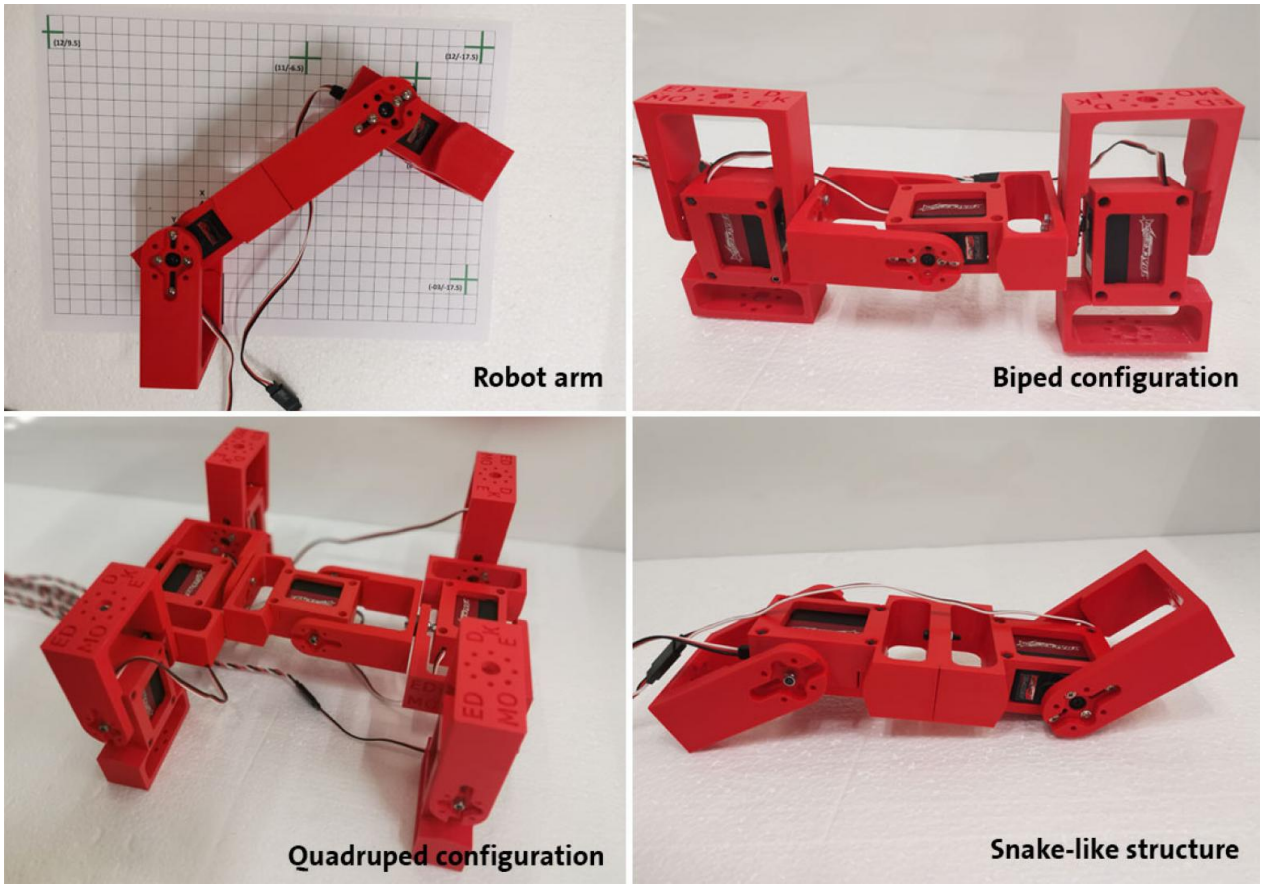


Figure 1.1: Different types of EDMO configuration [31]

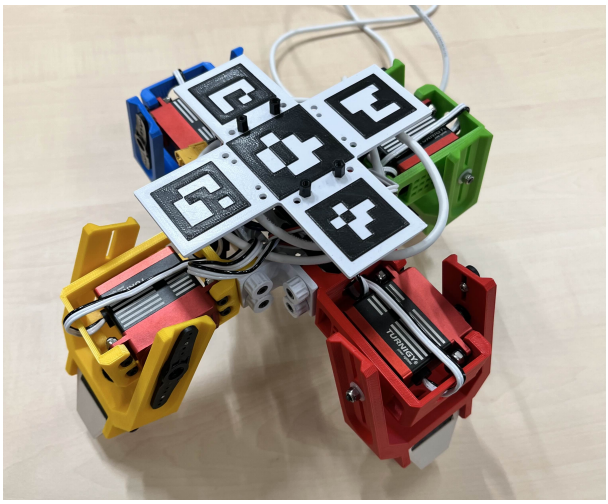
children in adjusting the robot's parameters to achieve an optimal gait, irrespective of the EDMO configuration.

To accomplish this, we will track the robot's movement, assess its performance based on various parameters, and provide individual feedback to support the children. To achieve these objectives, we will explore the following research questions:

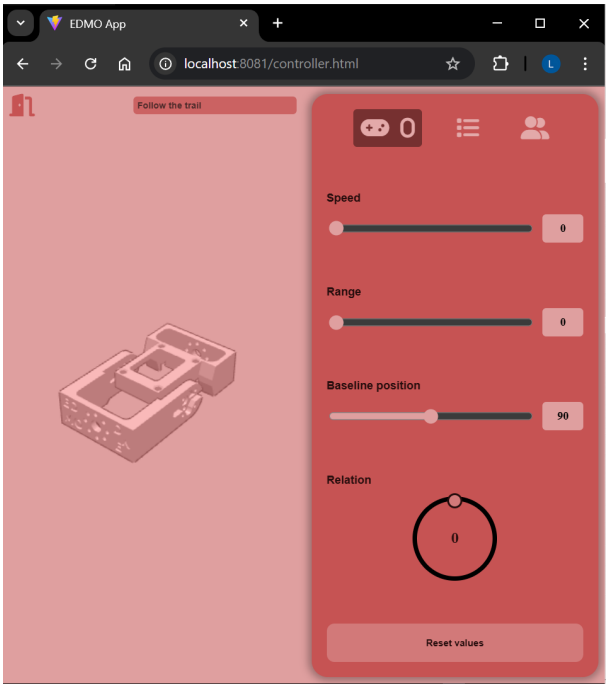
- How reliable are the GoPro camera and ArUco markers for accurately tracking the movement of the EDMO robot? (RQ1)
- What does the EDMO's speed-based fitness landscape look like? (RQ2)
- How do external factors, such as cables, hardware variations, and surface conditions, affect the reliability of the EDMO's movement? (RQ3)
- How reliably can gait parameters be transferred between EDMO robots? (RQ4)

3 EDMO

An EDMO robot consists of multiple interconnected EDMO modules. Each module (Figure 3.1) comprises two main components: a body containing a servo motor and a leg actuated by this motor. In the children's interactive interface (Figure 2.1b), the motion of each leg is described using four intuitive parameters—speed, range, baseline position, and relation—to facilitate ease of understanding and control. In the context of this research, these terms correspond to the standard Central Pattern Generator (CPG) terminology: frequency, amplitude, offset, and phase, respectively [13]. EDMO's locomotion system is based on a modified CPG model, in which all modules reference a common sine wave, rather than generating their own internal oscillations [28].



(a)



(b)

Figure 2.1: Study setup showing (a) the EDMO spider configuration and (b) the children's interface.

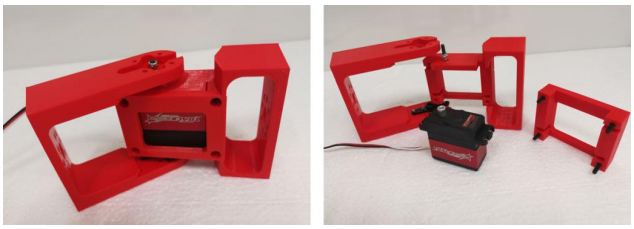


Figure 3.1: Structure of a single EDMO module

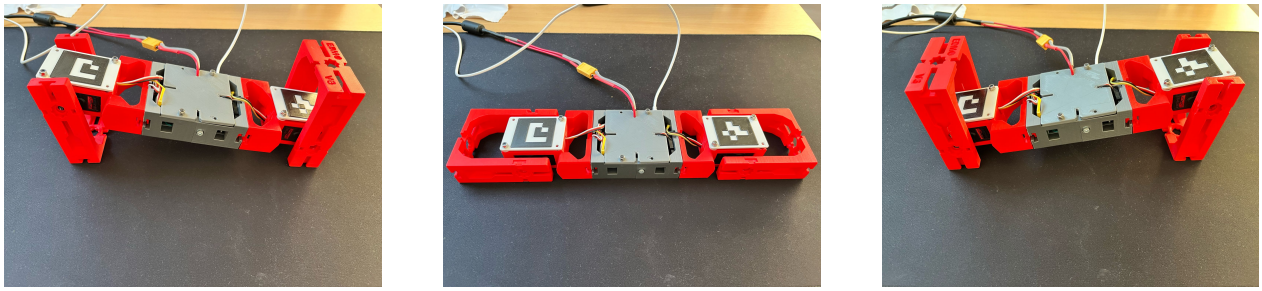
Within a given configuration, all modules share the same **frequency**, which controls the speed of the leg. This parameter is constrained to a dimensionless range from 0 to 1 and is the only one that supports decimal values, ensuring safe operation of the hardware. The remaining parameters are defined as follows:

- **Amplitude**: Specifies the range of leg motion, from 0° to 90°.
- **Offset**: Determines the baseline orientation of the leg, adjustable between 0° and 180°.
- **Phase**: Allows temporal shifting of the module's motion along the shared sine wave, within a range of 0° to 360°.

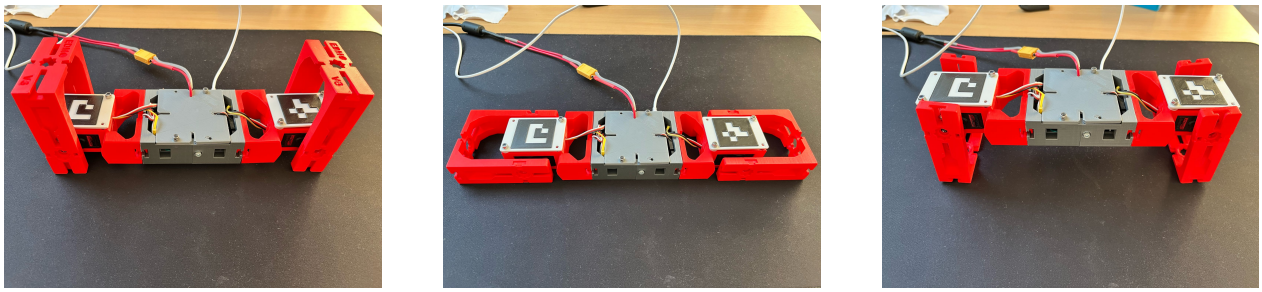
In this research, we will focus our experiments on the snake-like configuration, which consists of 2 EDMO modules (Figure 3.2). However, we have developed a framework that can be applied to any combination of EDMO modules. Throughout the rest of this paper, the term "EDMO" will refer specifically to the snake configuration.

4 Setup

Our setup, shown in Figure 4.1, includes a frame with an overhead camera and a table mat designed to provide a consistent surface for the EDMO to move on. The overhead camera ensures that the ArUco markers on both the mat and the EDMO remain consistently within its field of view. The mat features an ArUco marker at each corner, while each leg of the EDMO is equipped with an ArUco marker, as illustrated in Figure 4.1b.

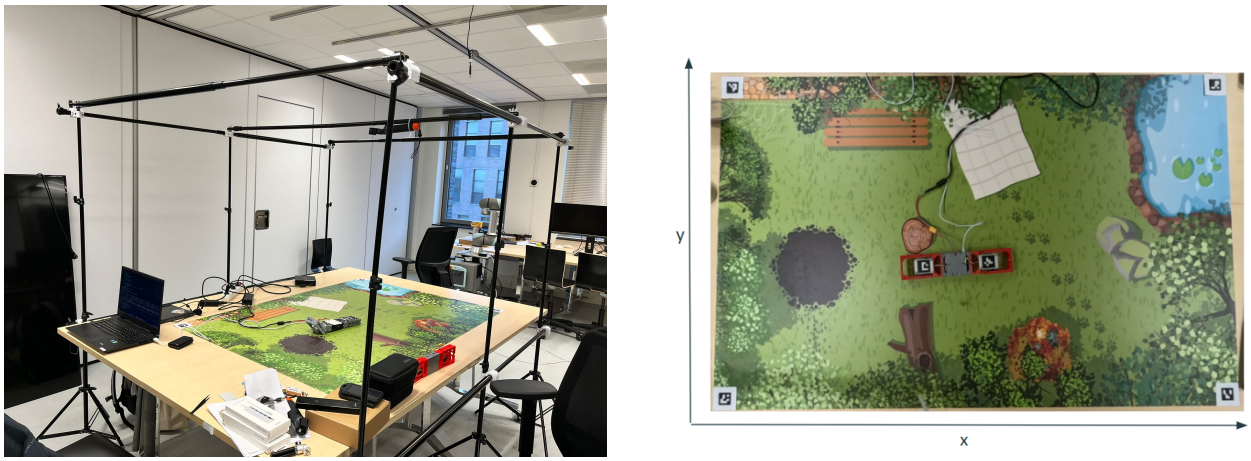


Parameters: frequency=1, amp1=amp2=90, off1=off2=90, phase1=0, phase2=0



Parameters: frequency=1, amp1=amp2=90, off1=off2=90, phase1=0, phase2=180

Figure 3.2: Snake-like configuration example set of parameters. A configuration of 2 modules is said to have 2 legs (ampK is the amplitude of leg K, offK is the offset of leg K, and phaseK is the phase of leg K).



(a) Frame setup

(b) Table mat with ArUco markers

Figure 4.1: Experimental setup showing (a) the frame setup and (b) the table mat with ArUco markers.

4.1 Tracking

To track the EDMO's movement, we used a GoPro HERO13 Black camera in combination with ArUco markers. A Logitech C920 HD Pro Webcam was initially used, but its image quality was not sufficient for reliably detecting the ArUco markers in our experimental setup.

GoPro

The experiments were recorded using a GoPro camera. To ensure reproducibility and accuracy in the analysis, it was crucial to maintain consistent settings. Given the variety of adjustable parameters on the GoPro, such as framing, resolution, frame rate, and lens options, we evaluated different combinations of these settings to determine the most suitable configuration for our application. The details of this exploration are discussed

in section 6.1.

GoPro Package To facilitate communication with the GoPro, we developed a GoPro package based on the code provided by OpenGoPro [9]. This package supports three distinct methods for interacting with a GoPro:

- **Bluetooth**: Bluetooth enables a device to establish a connection with the GoPro and send basic commands. It is primarily used to initiate communication, such as activating the GoPro's Wi-Fi or connecting the GoPro to an external network (Figure 8.1, 8.2).
- **Wi-Fi**: The GoPro can function as an Access Point (after enabling its Wi-Fi, as shown in Figure 8.1), allowing multiple clients to connect and interact using the OpenGoPro HTTP API [10]. This method provides significantly more functionality compared to Bluetooth and is the primary communication method employed in our experiments. A list of the implemented commands can be found in Table 8.1.
- **Network**: In this mode, the GoPro operates as a Station, connecting to an external Access Point such as a router or switch (illustrated in Figure 8.2). This setup enables control of multiple GoPros via the OpenGoPro HTTP API but requires the router to have an active internet connection.

ArUco markers

ArUco markers (Figure 8.3) are employed to detect the position and orientation of the EDMO. These markers are commonly used in computer vision applications due to their robustness, simplicity, and versatility [21]. Additionally, they are optimized for detection and pose estimation, making them highly suitable for our experiments. ArUco markers are available in various sizes and forms, and for our purposes, we chose the smallest 6×6 square markers, which feature a black border and a 4×4 pattern in the center. This option was selected because it provides the largest square size while accommodating the limited space available on the EDMO.

ArUco_Markers_Pose package We also implemented a package to calibrate our camera, detect and compute the position of objects having ArUco markers on it relative to fixed coordinate markers that represent the world frame, this was based on *Anton Sokolchenko* and *SowmiyaNarayanan G*'s work on ArUco pose estimation [26]. This package takes in a video as input and returns the position of the EDMO on each frame according to the coordinate system shown in Figure 4.1b where the z-axis points upward from the table to the camera. An example of the detected ArUco markers and an example of the tracked EDMO's position can be seen in Figure 8.4.

Calibration Camera calibration was performed using the OpenCV library [22], which computes the camera's calibration matrix and distortion coefficients by matching 2D image points to known 3D world coordinates of a checkerboard with a predefined square size [18]. This process involves capturing approximately 40 images from different positions and orientations.

The calibration matrix describes how 3D world points are projected onto the 2D image plane under the assumption of a pinhole camera model [32]. It is determined by intrinsic camera parameters such as focal length and optical centers.

The distortion coefficients, on the other hand, account for image distortions caused by lens imperfections. There are two main types of distortion:

1. Radial distortion occurs when light rays bend more significantly as they move away from the optical center (see Figure 8.5). This effect is modeled using a polynomial in r, the radial distance from the optical center. The equations [22] are:

$$x_{\text{distorted}} = x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right)$$ [1]
$$y_{\text{distorted}} = y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right)$$

   The parameters k_i are estimated by aligning the distorted 2D image points($x_{\text{distorted}}$, $y_{\text{distorted}}$) with the corresponding 3D checkerboard points(x, y).

[1] More details on the equations can be found in [4, p. 375]

2. Tangential distortion arises when the camera sensor is not perfectly parallel to the lens. This misalignment causes the image to distort asymmetrically(see Figure 8.6). The effect is modeled by considering the interaction between the x and y coordinates. The equations [22] are:

$$x_{\text{distorted}} = x + 2p_1xy + p_2(r^2 + 2x^2)$$
$$y_{\text{distorted}} = y + p_1(r^2 + 2y^2) + 2p_2xy$$

Similar to radial distortion, the parameters p_i are estimated by fitting the positions of points on the checkerboard pattern(x, y) to the points in the captured images($x_{\text{distorted}}$, $y_{\text{distorted}}$).

The accuracy of calibration depends on the quality of the images used. The estimated distortion parameters can vary significantly depending on the checkerboard's positioning and orientation across different images. A key challenge is selecting images where all checkerboard corners are detected correctly, as even minor pixel deviations can affect the calibration and they are difficult to identify visually. While reprojection error, which measures the difference between reprojected 2D points and their actual 3D positions, can help assess calibration quality, it is only reliable if the initial corner detections are accurate.

5 Parameter space exploration

To effectively guide the children, it is necessary to understand the fitness landscape and identify the optimal set of parameters as it reveals which combinations lead to successful robot movement and allows us to provide meaningful and goal-oriented feedback. In this context, we define successful movement as the robot achieving visible displacement in a consistent direction, measured by the robot's speed. To achieve this, we opted for optimization algorithms, which are widely used for modular robots due to their numerous advantages. First, these algorithms do not rely on a simulator or model, eliminating the challenges associated with transferring parameters from simulation to the real world. Second, they can accommodate any new robot configuration and are capable of learning new gaits. Lastly, these algorithms allow the robot to adapt to changes in the environment, enhancing its flexibility and performance.

When selecting the optimization algorithm, the most important aspects were that the algorithm should not require gradient information, should not be computationally intensive, and should converge quickly to ensure efficient exploration of the parameter space. For example (Figure 5.1), gradient-based methods (e.g., Stochastic gradient descent (SGD) [33], Adaptive Moment Estimation (Adam) [16]) are unsuitable because gradient information is unavailable in our real-world experimental setup. Similarly, Genetic Algorithms are impractical due to the large number of evaluations they require, which is prohibitive when experimenting with physical robots in real time. On the other hand, algorithms like Bayesian Optimization and Powell's method are well-suited because they are derivative-free, computationally efficient, and capable of finding good solutions within a reasonable number of evaluations. In the end, Powell's method was chosen because its one-parameter-at-a-time exploration strategy produces a clear and interpretable path through the parameter space, making it especially suitable for helping children understand how individual parameters influence the robot's movement. Further details about its application and advantages are discussed in the next paragraph.

Algorithm	**Gradient**	**Computationally intensive**	**#Evaluations**	**Real-World Suitability**
SGD / Adam	Yes	No	Moderate	No
Genetic Algorithm	No	Yes	High	No
Bayesian Optimization	No	Moderate	Low–Mod.	Yes
Powell's Method	No	No	Low	Yes

Table 5.1: Comparison of optimization algorithms for physical robot experiments

Powell's method [25, p. 415] Powell's method is a fast, heuristic optimization algorithm designed for scenarios where gradients are unavailable. It works by performing successive line minimizations along a set of orthogonal directions, systematically exploring the parameter space one dimension at a time (example in Figure 8.7). The algorithm begins by selecting a random starting point in the parameter space, and for our application, we initialize the orthogonal directions using the different parameters. Powell's method is used to find the minimum of a function, we use this approach to minimize the inverse of the EDMO's speed, effectively maximizing its velocity.

This systematic exploration strategy is a key advantage of Powell's method, as it creates a clear and interpretable path through the parameter space. By optimizing one parameter at a time, we can guide the children through the optimization process in a structured and intuitive way, helping them understand the effect of each change. Unlike other optimization algorithms, which may jump unpredictably through the parameter space, Powell's method provides a deterministic and methodical approach. This ability to systematically explore and visualize the impact of individual parameters aligns perfectly with our goal of making the optimization process both accessible and educational for the children.

Powell's method is particularly appropriate for our application, as it is designed to find local optima—precisely the objective of our optimization process. Our goal is not to converge on a single global maximum, but rather to identify multiple local optima across the parameter space. This enables us to guide children toward nearby, effective solutions that emerge from their own exploration, making the learning experience more personalized and comprehensible. Since multiple solutions can result in successful locomotion, targeting a single global optimum is both unnecessary and potentially limiting. To better understand the structure of the fitness landscape, we also analyze the sensitivity of each parameter by exploring the vicinity of the identified optima. After reaching a local maximum, we apply small random perturbations (within $\pm$ 10) to the parameters to encourage further exploration.

Parameter Space The snake EDMO has two legs and seven different parameters, but not all of these parameters are relevant for exploration:

- Frequency: Through numerous trials, we observed that for the majority of parameter combinations, higher frequency result in faster movement. While there are a few specific cases where this does not hold, they are rare. To streamline our exploration and reduce the parameter space, we opted to fix the frequency at its maximum value for all the experiments.
- Phase: The individual phases of the two legs are not meaningful by themselves; only the difference between their phases provides useful information. Therefore, we will focus solely on the phase difference during the experiments.
- Symmetry: Due to the symmetrical nature of the EDMO, a phase difference of 90° or -90° results in the same movement, but in opposite directions. Similarly, a phase difference of 270° is equivalent to -90°, which is functionally identical to 90° in terms of movement speed. Thus, we limit our exploration of phase differences to the range of 0° to 180°.

With all those simplifications we end up with a 5 dimensions parameter space, namely the amplitude of leg 1 and 2 (0-90), the offset of leg 1 and 2 (0-180) and the phase difference with a reduced search range (0-180).

6 Experiments and Results

Experiments were conducted on two EDMOs with the snake configuration, that we will call Snake 1 and Snake 2, in order to answer the research questions (RQ1-RQ4).

6.1 Tracking reliability

To evaluate the reliability of our tracking, we use two metrics to assess calibration error and camera noise:

- Static distance error: Given the known positions of the ArUco markers on the mat and their exact distances from each other, we compute the difference between the actual distance and the estimated distance for each camera frame. This difference is expressed as errors along the x, y, and z axes (as defined in Section 4.1), providing insights into the maximum distortion in the distance measurement on the mat.
- Dynamic distance error: We performed experiments by placing a ruler on the mat at different positions and orientations (Figure 6.1). The EDMO was then manually moved along the ruler over a fixed distance—108 cm horizontally and 72 cm vertically, 3 times and 2 times the length of the EDMO respectively. By comparing the known distance with the computed distance, we assessed the accuracy and reliability of our tracking system in detecting the EDMO's movement.

GoPro settings

We experimented with various GoPro settings to determine the most suitable configuration for our needs. Certain settings were specifically chosen, such as the framing, where we opted for a widescreen format with a 16:9 aspect ratio. This format was ideal as it aligned well with the mat layout, while other aspect ratios did not provide additional benefits for pose estimation. We also selected a frame rate of 30 FPS, the lowest available, since the EDMO's movement speed did not necessitate a higher frame rate. Increasing the frame rate would have only added extra load to the data transfer without improving detection performance.

We conducted tests using two different profiles (Standard and HDR), two resolution options (1080p and 4K), two digital lenses (Linear and Wide), and with or without HyperSmooth, which stabilizes the video frames. The results are summarized in Table 6.1.

Table 6.1: Results of GoPro settings experiment (Distribution plots in Figure 8.8)

Profile	Resolution	Digital Lens	HyperSmooth	Static error (cm)			Dynamic error (cm)	
				x	y	z	x	y
Standard	1080	Wide	Off	2.9 ± 4.9	-10.3 ± 1.7	37.9±14.0	±4.5	±2.9
Standard	1080	Wide	On	5.0 ± 0.5	1.7 ± 2.0	-4.7 ± 5.4	±4.4	±2.9
Standard	1080	Linear	On	6.8 ± 0.9	3.5 ± 1.2	$\mathbf{-1.1 \pm 1.1}$	±12.4	±3.2
Standard	4K	Wide	On	$\mathbf{-0.8 \pm 1.6}$	0.9 ± 1.7	-1.6 ± 1.7	**±1.3**	**±0.8**
HDR	4K	Wide	On	2.3 ± 0.3	$\mathbf{0.6 \pm 0.5}$	-19.8 ± 2.4	±1.8	±1.6
HDR	4K	Linear	On	3.8 ± 0.3	0.5 ± 1.0	-1.1 ± 2.4	±2.5	±1.2

Results and discussion

The key difference between the digital lenses lies in the amount of distortion they introduce. The Linear lens produces minimal distortion but requires the overhead camera to be placed higher (1 meter from the mat) to capture the entire area. In contrast, the Wide lens introduces more distortion but allows the camera to be positioned closer to the mat (88 cm). Despite the higher distortion of the Wide lens, its closer proximity to the mat enables more accurate calibration, which effectively corrects the distortion and results in a lower dynamic error

Additionally, while the Linear lens demonstrates greater accuracy on the z-axis (depth perception), our primary focus is on tracking the movement of the EDMO in the x and y-axes, making the Wide lens more suitable for our application. Another notable observation is that the x-axis error is consistently greater than the y-axis error, which is expected due to the longer length of the x-axis, leading to a higher accumulation of errors over its span.

In summary, the Wide lens, despite its higher distortion, is better suited for our needs due to its closer placement, more accurate calibration, and superior performance in tracking movement along the x and y-axes.

As shown in Table 6.1, the best overall performance was achieved using the Standard profile with 4K resolution and HyperSmooth enabled. However, processing 4K video incurs a significant computational cost, requiring 20 times more time and memory than 1080p video. For the 1080p settings, the optimal performance was observed with the Wide lens and HyperSmooth enabled. These settings were selected for our experiments, as they offer high-quality visual data while minimizing the computational load during data analysis.

Our findings effectively address *RQ1: How reliable are the GoPro camera and ArUco markers for accurately tracking the movement of the EDMO robot?* Using the Standard profile with wide lens, 1080p resolution and HyperSmooth enabled, the tracking of EDMOs demonstrates an good accuracy, with an error margin of ± 4.4cm on the x-axis and ± 2.9cm on the y-axis. While having the possibility to use a 4K resolution to attain a higher accuracy with an error margin of ± 1.3cm on the x-axis and ± 0.8cm on the y-axis.

6.2 Fitness landscape

To explore the fitness landscape, we defined the robot's speed as the objective function. In order to compute the EDMO's speed, we tested two methods. The first measured the distance traveled between the first and last frames. While this approach was sensitive to noise in those specific frames, it provided the most

Figure 6.1: Experimental setup for testing the GoPro settings

consistent measurements. The second method involved calculating the distance traveled between successive frames and averaging the results. However, this approach tended to accumulate small errors across frames, leading to highly noisy and unreliable measurements (see Figure 8.9).

Initially, we attempted a systematic search of the parameter space to map the fitness landscape. However, this strategy quickly proved impractical. Due to the vastness of the parameter space, only a small subset of configurations yielded viable gaits, rendering exhaustive search inefficient. Moreover, conducting experiments on the physical robot is time-intensive, and many parameter combinations resulted in the robot colliding with the table—posing a risk of hardware damage if repeated. For these reasons, we adopted Powell's method to perform a more efficient and targeted exploration of the search space.

Powell's method

Powell's method converges rapidly but is highly sensitive to initial conditions, making it prone to getting stuck in local maxima. To address this limitation, we introduced randomness to explore the surrounding search space. When the method converges and remains at the same point for more than one iteration—indicating a potential local maximum—we randomly modify the parameters within the range [0, 10], this operation is done for the first convergence of an iteration. The range [0, 10] was selected to achieve a balance between introducing enough randomness to avoid being dominated by external factors, discussed in Section 6.3, and ensuring that the perturbations are not excessively large, allowing the search to remain focused on exploring the vicinity of the local maximum. Additionally, we implemented a dictionary to store previously visited positions, enhancing search efficiency and systematically recording the performance of all evaluated parameter sets. This approach allows us to explore the search space more effectively while maintaining a record of past evaluations to avoid redundant computations.

Each iteration of Powell's method involves a line minimization along each of the five dimensions in our search space, with an average runtime of approximately 17 minutes per iteration. On average, the algorithm converges after four iterations. By selecting random starting points, global optima can be reached in as few as three runs (±3.4 hours).

The experimental procedure follows the framework illustrated in Figure 6.2, where Powell's method iterates through line minimization using the golden-section search method [25].

Results and discussion

The complete fitness landscapes for Snake 1 and Snake 2 are presented in Figures 8.10 and 8.11, respectively. In these figures, "amp" denotes the amplitude, "off" the offset, and "phb_diff" the phase difference between legs 1 and 2. To visualize the five-dimensional fitness landscape, we use a series of 2D heatmaps, each illustrating the relationship between pairs of parameters. Each parameter in the heatmaps is plotted with a resolution of 60, meaning each rectangle represents a range of values[1]. This resolution balances the limited number of explored parameters with readability.

For our analysis, we focus on four specific plots to gain insights across all five parameters: *amp1 vs amp2*, *amp1 vs off2*, *off1 vs off2*, and *off1 vs phb2*. From these plots, we derive the following observations:

[1]For example, a parameter with a range from 0 to 180 would have each rectangle representing 3 values (180/60).

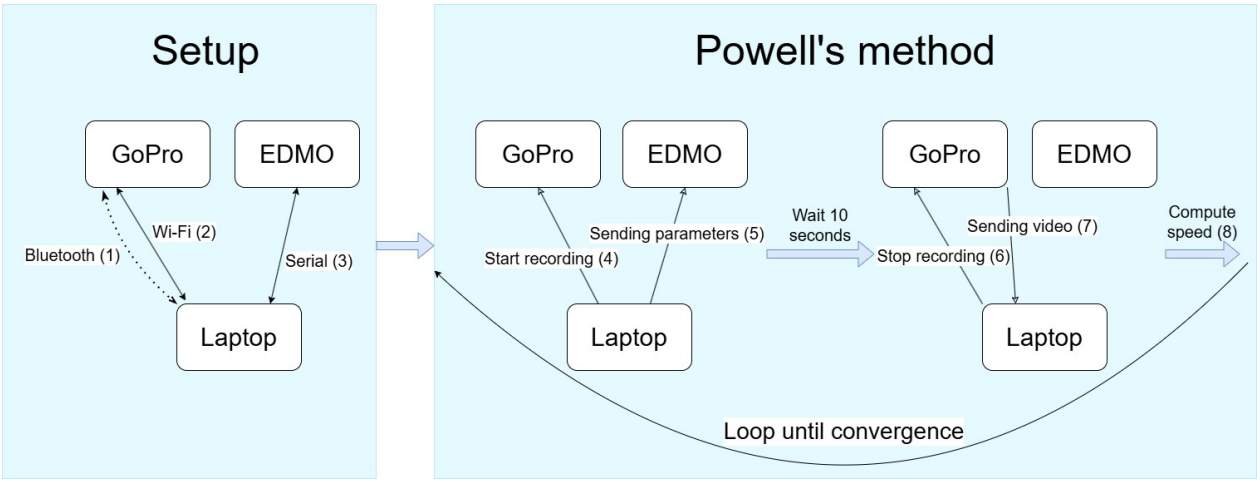


Figure 6.2: Experiment pipeline

- **Optimal regions in the fitness landscapes:** In the fitness landscape plots, Snake 1's *off1 vs off2* plot reveals three distinct regions with optimal speeds, which we will refer to as the global optima, highlighted in Figure 6.3a. Similarly, Snake 2's *amp1 vs amp2* plot shows three optimal regions (Figure 6.3b). Although Snake 2's optimal regions appear broader, its maximum speed is approximately 12% lower than Snake 1's due to motor overheating issues (detailed in Section 6.4). This reduced speed range can make some gaits appear more effective than they are, as the speeds of optimal and non-optimal gaits are closer. The highlighted optima in Figure 6.3 can be traced to corresponding optima in the rest of the fitness landscape. For Snake 1, these are highlighted in Figures 6.7, 6.8, and 6.9. For Snake 2, they are highlighted in Figures 6.10, 6.11, and 6.12.

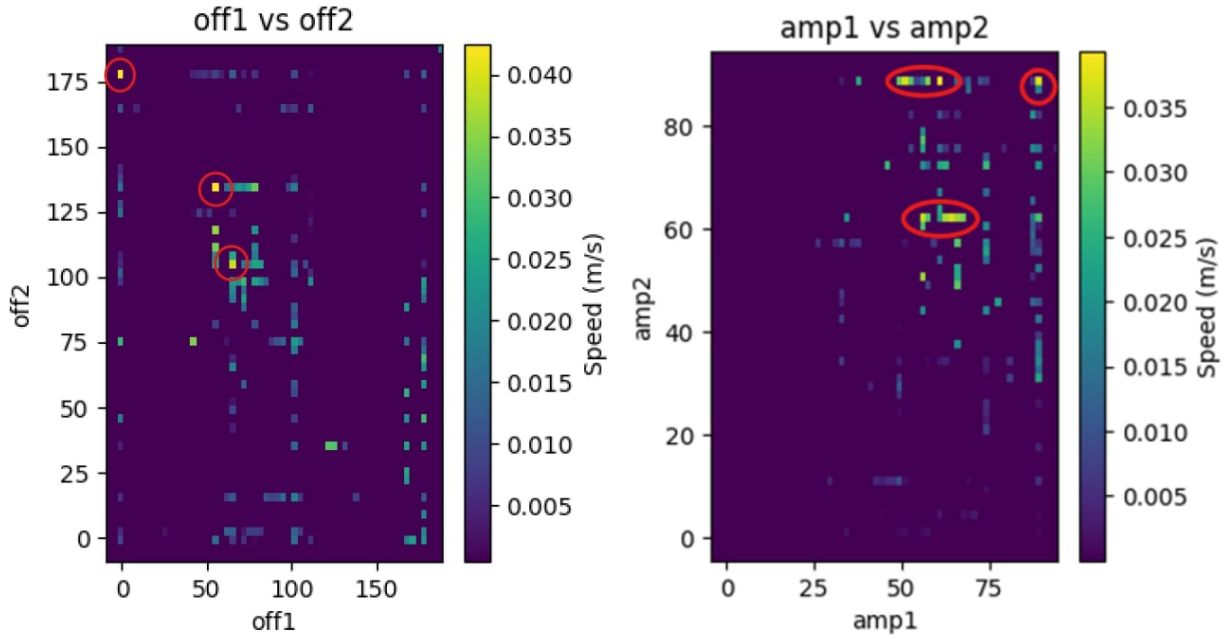
- **Optimal Gaits:** Corresponding to the global optima, we identified three types of optimal gaits: "snake-like," "caterpillar-like," and "walk-like," depicted in Figures 6.4, 6.5, and 6.6, respectively. These gaits were identified in Snake 1. However, for Snake 2, only the "snake-like" and "caterpillar-like" gaits were optimal; the "walk-like" gait was not achievable (explained in Section 6.4). Although three distinct optimal regions are visible in the *amp1 vs amp2* plot for Snake 2, two of these regions correspond to a "snake-like" gait. The values of the other parameters, except for amp1, are very similar, as seen in Figures 6.11 and 6.12.

- **Comparison of Fitness Landscapes:** When comparing the fitness landscapes of both snakes side by side, we observe significant differences in the optimal regions, despite some overlapping areas. For instance, the "caterpillar-like" gaits are achieved with similar parameters for both snakes, yet Snake 2's speed is considerably lower. This indicates that certain gaits might be transferred between different EDMOs, although it remains unclear which gaits can be transferred and how easily this can be accomplished (more details in Section 6.4).

Furthermore, the complete exploration plots in Figures 8.12 and 8.13 depict the paths taken by Powell's algorithm during the exploration process. These plots help us understand the ease of reaching each local optimum within the parameter space and the sensitivity of these optima. For clarity, we use the same four plots as before, presented in Figures 6.13 and 6.14 for Snake 1 and Snake 2, respectively.

The general observation is that in the *amp1 vs amp2* plot, the exploration paths tend to move toward the top right corner, where both amplitudes are higher, which is an expected result. In the *off1 vs off2* plot, the paths generally move toward the middle range of both parameters. Except for Snake 1, some paths are drawn toward an optimum in the top left corner, corresponding to the "walk-like" gait which is not observed in Snake 2's *off1 vs off2* plot.

We also observe that some exploration paths get stuck in local optima far from the identified global optima. For example, this occurs in the bottom right corner of the *off1 vs off2* plot for Snake 1 and the top right corner of the *off1 vs phb_diff* plot for Snake 2. These local optima differ significantly between the two EDMOs, but the key takeaway is that while some exploration paths can be trapped in local optima, they are few in number. In most cases, the exploration paths converge toward the global optima regions, which



(a) Local maxima for Snake 1

(b) Local maxima for Snake 2

Figure 6.3: Local optima for both Snake edmos

is beneficial as it allows us to identify the general trend of where the global optima are located with only a few runs, thus narrowing the search to those areas. This observation holds for the Snake configuration but needs further investigation for other EDMO configurations.

We would have liked to study the sensitivity of the global optima, such as the size of the region they cover and the proximity required to be attracted to them. However, due to the observations in Section 6.3, the results obtained during this research can only provide a general trend. The noise generated by external factors and camera tracking makes it currently impossible to obtain precise measurements.

This effectively answers our *RQ2: What does the EDMO's speed-based fitness landscape look like?*, we observed 3 global optimas characterized by 3 optimal gaits which are the "snake-caterpillar-walk like" gaits, however Snake 2 can only reach the "snake-like" and "caterpillar-like" gaits. We have also identified that there are regions of local optima that are far from the global optima that can trap the exploration paths. Nevertheless, most exploration paths still converge toward the global optima.

Figure 6.4: Snake like optimal gait (amp1=68, amp2=48, off1=68, off2=108, phb_diff=68)

Figure 6.5: Caterpillar-like optimal gait (amp1=59, amp2=58, off1=57, off2=137, phb_diff=73)

Figure 6.6: Walk-like optimal gait (amp1=amp2=90, off1=0, off2=180, phb_diff=52)

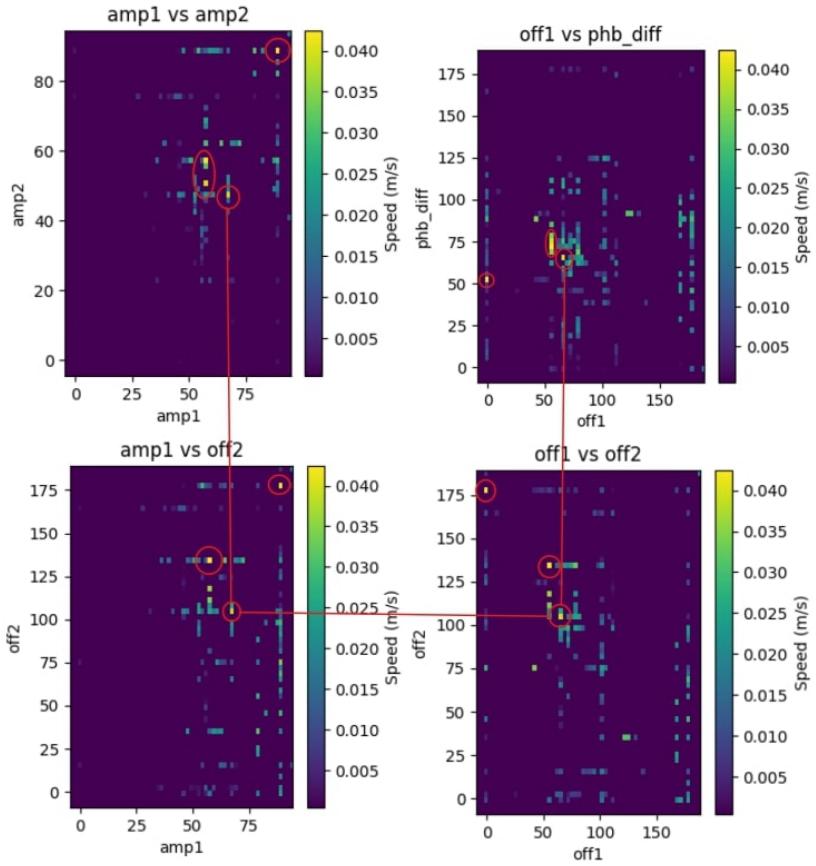


Figure 6.7: Snake 1 fitness landscape: "snake-like" optima

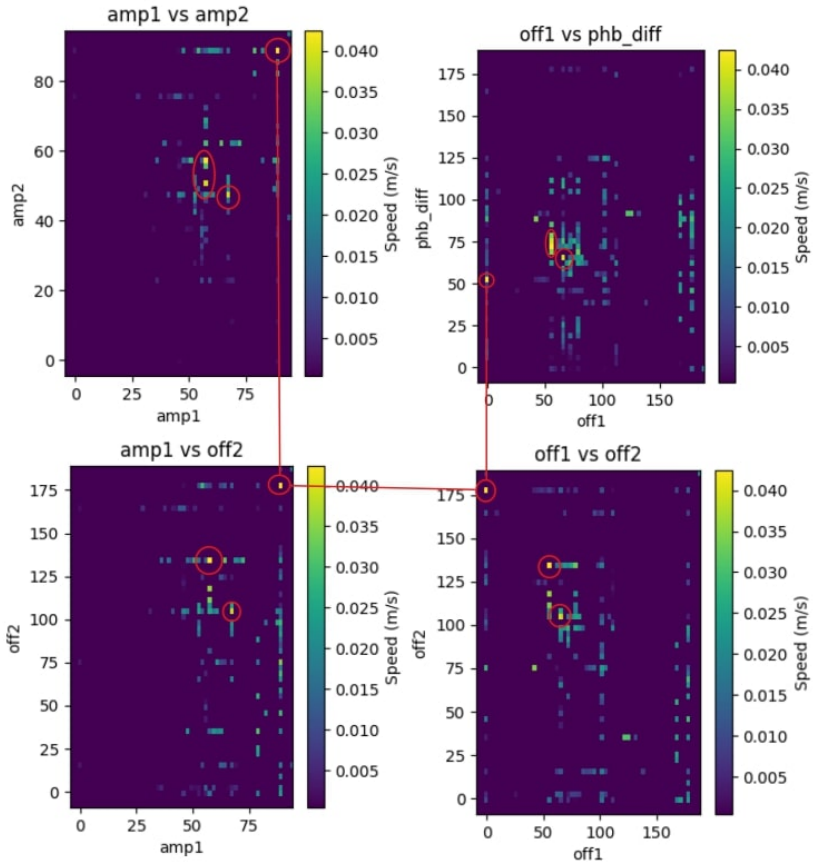


Figure 6.8: Snake 1 fitness landscape: "walk-like" optima

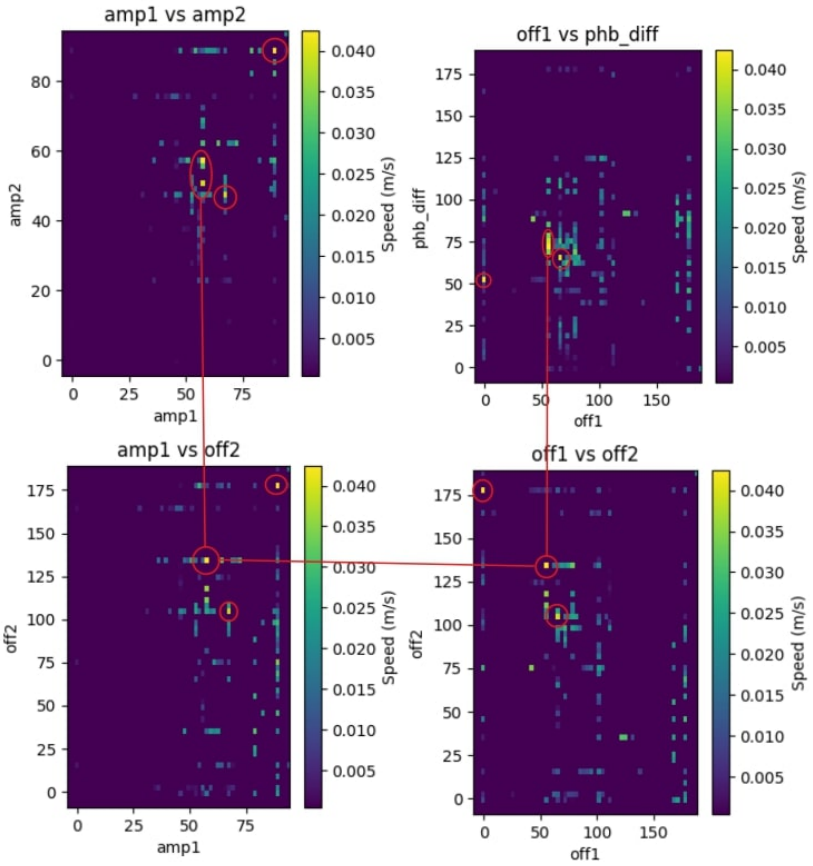


Figure 6.9: Snake 1 fitness landscape: "caterpillar-like" optima

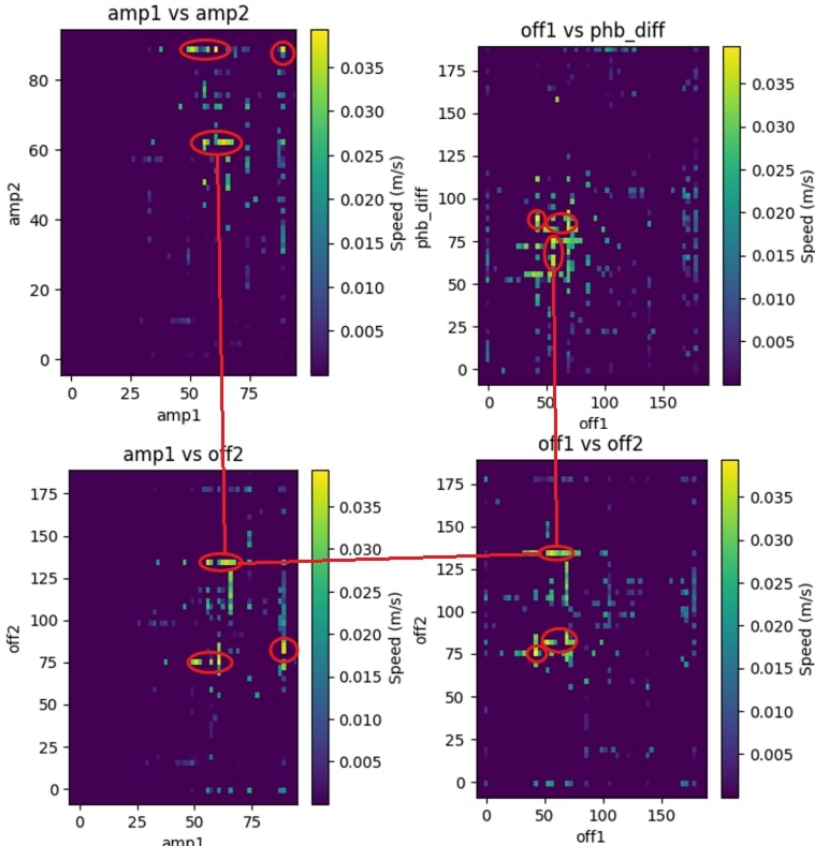


Figure 6.10: Snake 2 fitness landscape: "caterpillar-like" optima

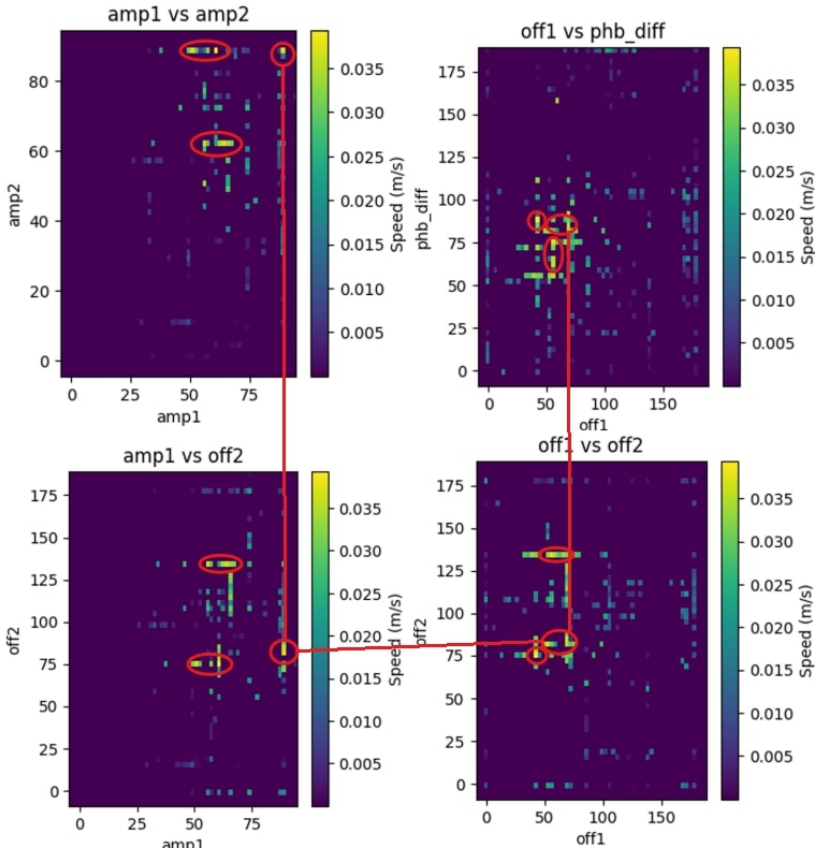


Figure 6.11: Snake 1 fitness landscape: "snake-like" optima

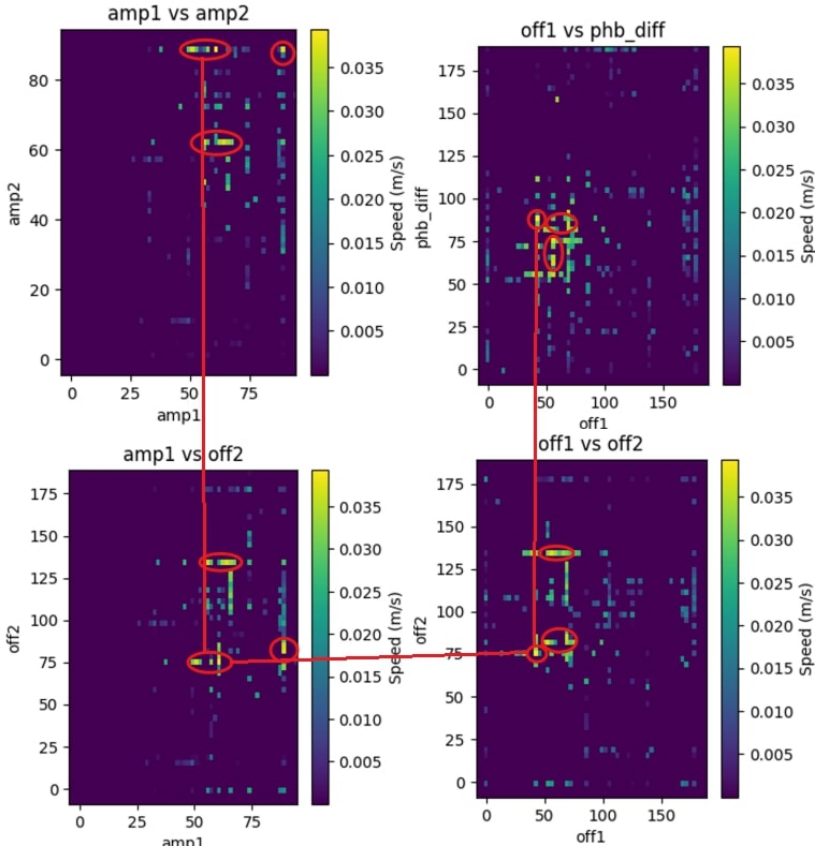


Figure 6.12: Snake 2 fitness landscape: "snake-like" optima

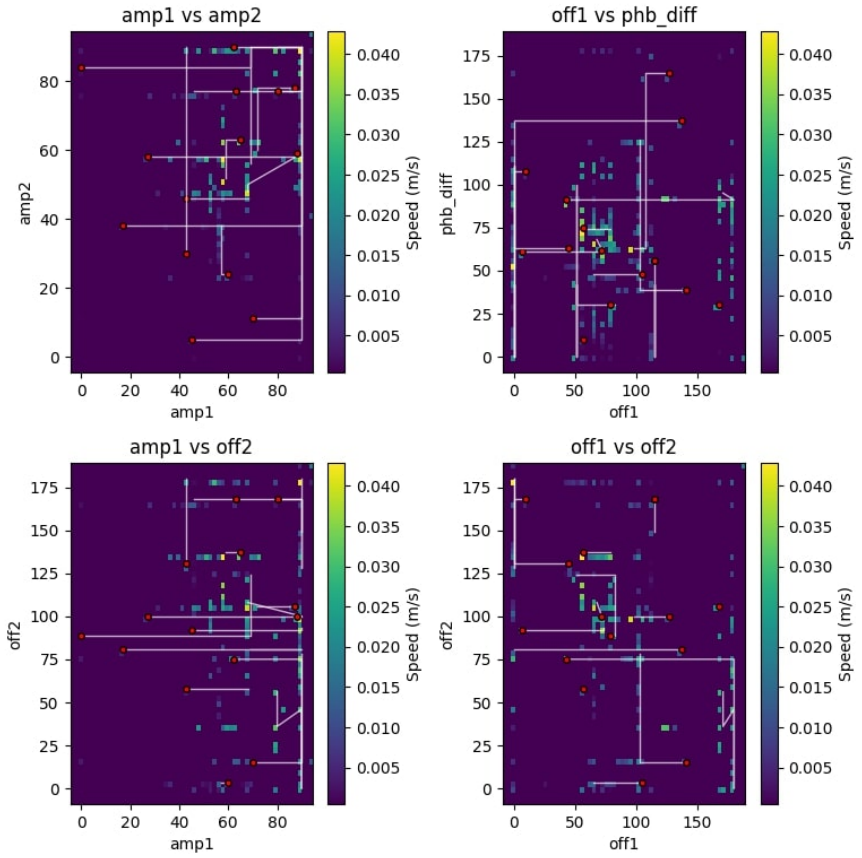


Figure 6.13: Snake 1 exploration paths

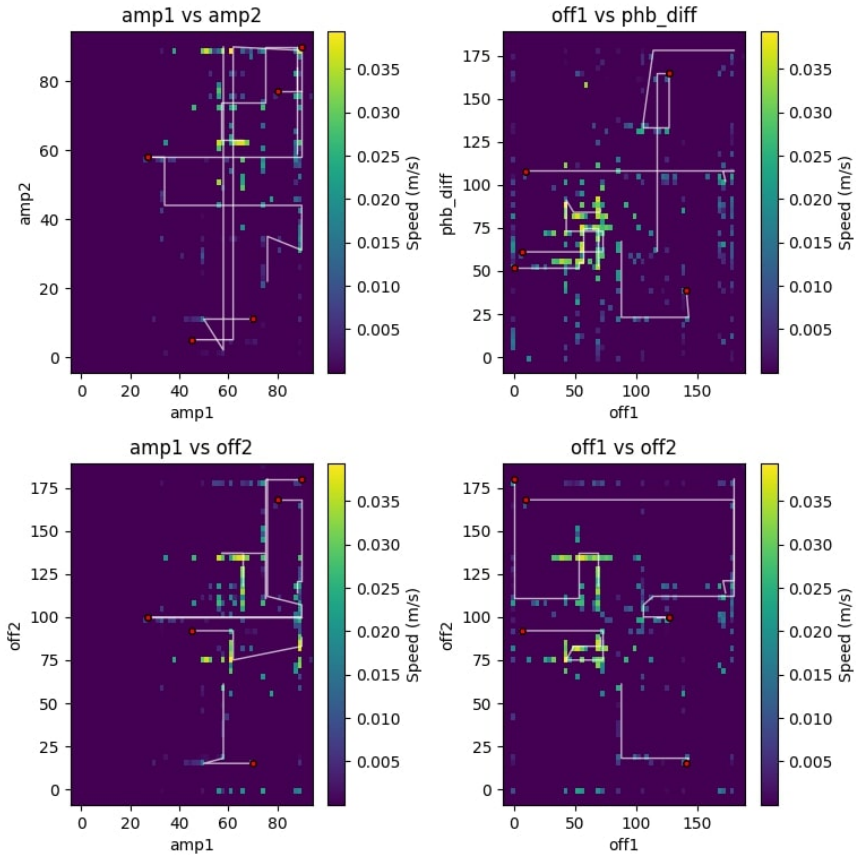


Figure 6.14: Snake 2 exploration paths

6.3 Impact of external factors

To assess the influence of external factors such as cables, hardware, and surface conditions, we conducted experiments isolating each variable. For each scenario, we performed 15 runs using the optimal gait determined by Powell's method.

- Benchmark:
  As a benchmark to evaluate the results of the following experiments, we conducted 15 runs with the cables taped to the table. The robot was placed in the exact same starting position for each run to ensure consistent grip on the mat and minimize variations caused by cable influence. This setup, as shown in Figure 6.15, ensured that any variations observed were solely due to the hardware and motors.

Figure 6.15: Benchmark

- No cable:
  We conducted 15 consecutive runs at various locations on the mat, during which we manually lifted the cables to ensure they remained loose and minimally impacted the EDMO, as shown in Figure 6.16. This experiment aimed to assess how variations in grip at different spots on the mat affected the EDMO's movement.

Figure 6.16: Holding the cables

- Cable:
  We conducted 15 consecutive runs at different locations on the mat, allowing the cables to remain on

the table, as they would during a regular EDMO run.

Results and discussion

The results, presented in Figure 6.17, effectively answers *RQ3: How do external factors, such as cables, hardware variations, and surface conditions, affect the reliability of the EDMO's movement?*.

The data clearly show that the speed is most consistent (with the shortest inter-quartile range) in the benchmark scenario, highlighting that the motors are the most stable component in our experiments. In the "no cable" experiment, we observe a high interquartile range for Snake 1 but a lower one for Snake 2. This discrepancy can be attributed to the variability introduced by the human factor—specifically, the person holding the cable—which is challenging to quantify and control.

Furthermore, we note a significant increase in speed when the cables are removed, which aligns with our expectations. This highlights the substantial influence cables have on the EDMO's performance. For Snake 1, the presence of cables leads to a 33% reduction in speed, while for Snake 2, the decrease is nearly 50%. These experiments were conducted using optimal gaits, implying that the detrimental effect of cables would be even more pronounced for less efficient gaits. Suggestions for mitigating the impact of external factors are discussed in Section 7.2.

6.4 Parameter transfer between 2 EDMOs

Before attempting to transfer the optimal gait parameters from one EDMO to the other, several observations were made. As shown in Figure 6.18a, both Snake 1 and Snake 2 were set to the same initial position with the same parameters (frequency 0, amplitude 0, offset 90, and phase 0). However, their positions differed, and the same discrepancy was observed when adjusting the amplitude to its minimum and maximum values, as shown in Figure 8.14. To align the positions of both snakes, an offset of -1 and +25 degrees was applied to Snake 1, as shown in Figure 6.18b. This misalignment is a result of differences in servo motor calibration, which makes it impossible to directly transfer gaits from one EDMO to the other. Consequently, the optimal "walk-like" gait (Figure 6.6) is unattainable for Snake 2 due to this calibration issue.

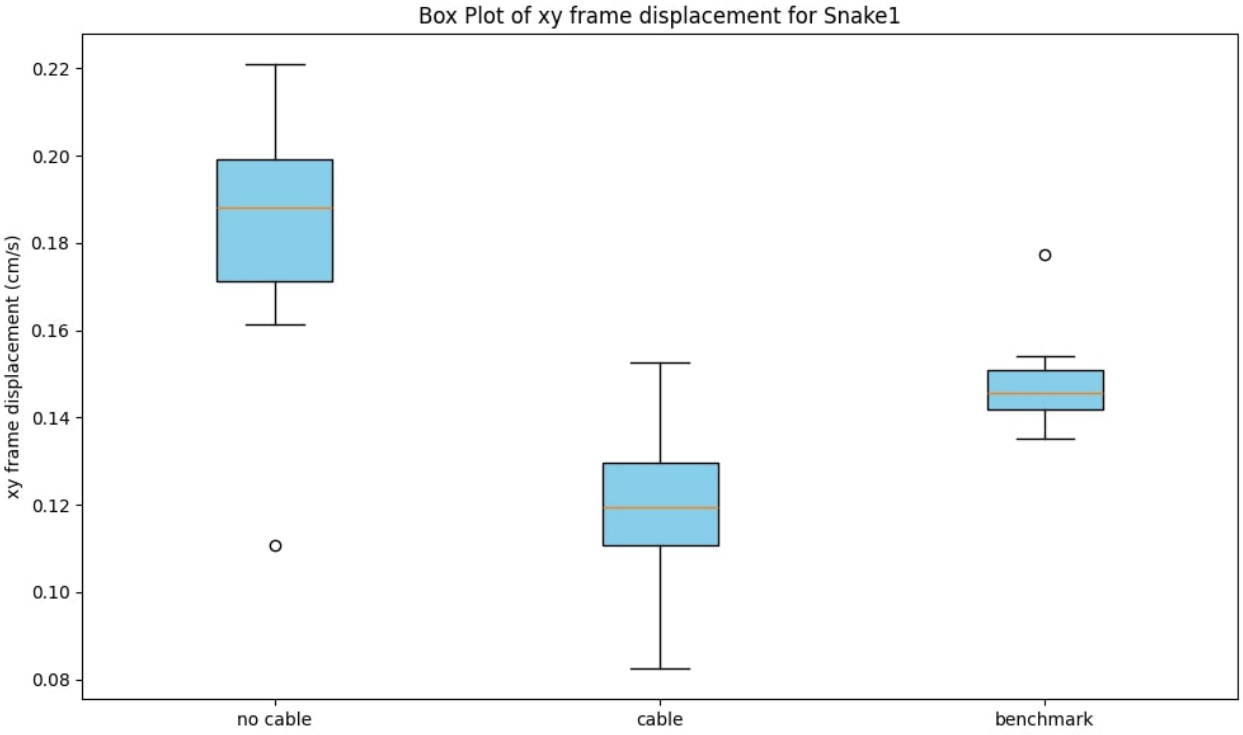
Additionally, one of Snake 2's servo motors tends to overheat, resulting in reduced torque. While we did not have the proper equipment to measure this torque, we observed that Snake 2's fitness landscape (Figure 8.11) showed lower speed values compared to Snake 1's, further suggesting the motor's weakened performance.

As observed in Section 6.2, the "caterpillar-like" gait yielded similar parameter values for both snakes. However, transferring these parameters between the snakes resulted in significantly different speeds due to the aforementioned issues, making it challenging to assess the success of the gait transfer.

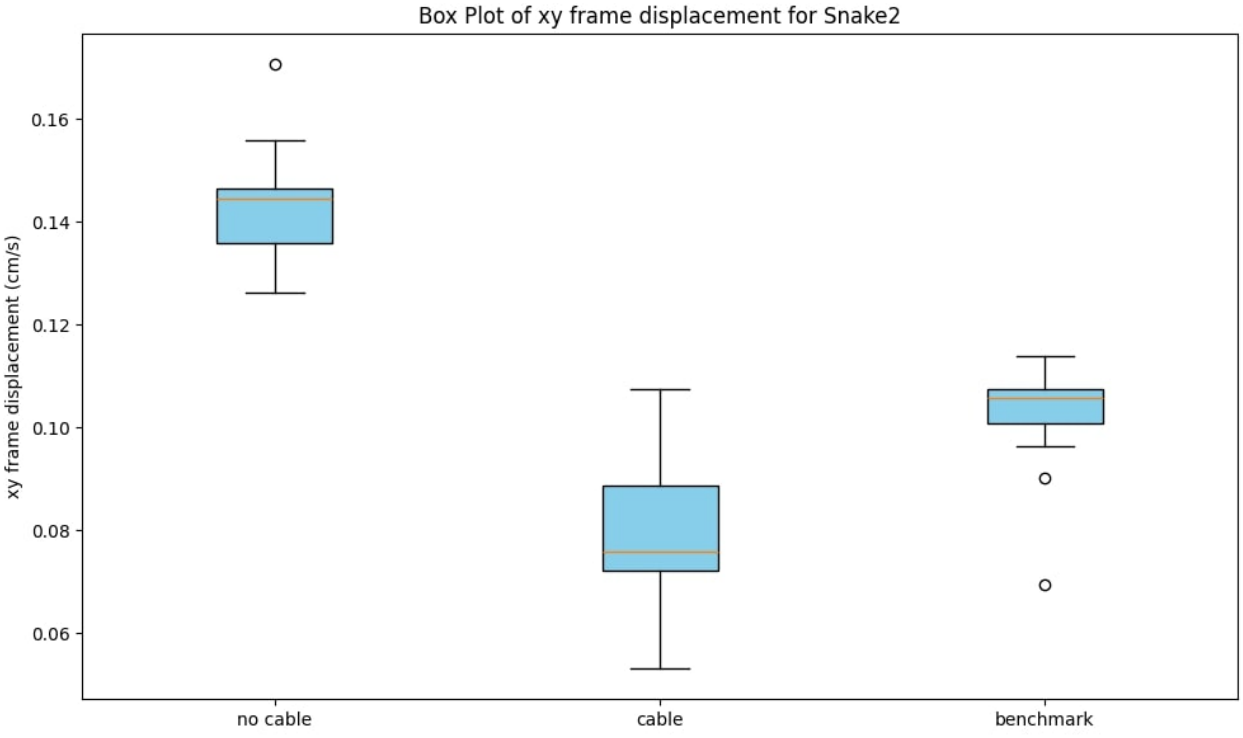
These observations indicate that the answer to *RQ4: How reliably can gait parameters be transferred between EDMO robots?* is that direct transfer is not feasible. Inherent differences in motor calibration and servo performance across EDMOs introduce significant variability, making it difficult to achieve consistent gait behavior. To address this limitation, future work should focus on two key improvements:

- Torque Measurement Implementation: Direct motor torque measurement would provide quantitative data about power output variations between different units, enabling systematic comparisons.
- Automated Calibration System: While current servos are manually calibrated with hard limits (180°) to prevent mechanical damage, this process could be automated:
  1. Determine extreme reachable positions through torque measurements
  2. Automatically set safe operational limits
  3. Implement a standardized gait transfer protocol:
     - Test identical parameter sets on multiple EDMOs
     - Measure and compare torque outputs
     - Calculate and apply necessary calibration offsets

  This automated approach would maintain hardware safety while enabling consistent performance across different EDMOs.

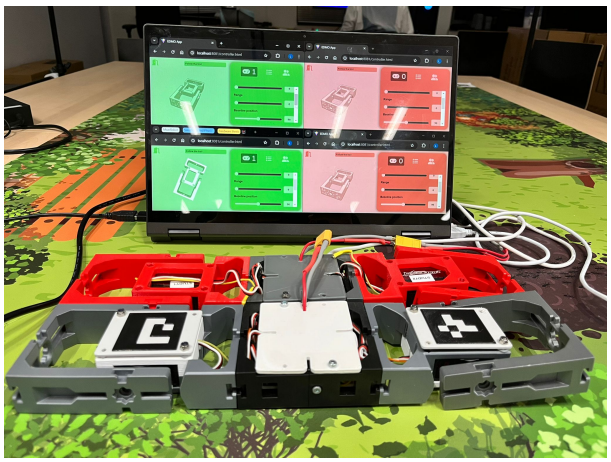


(a)

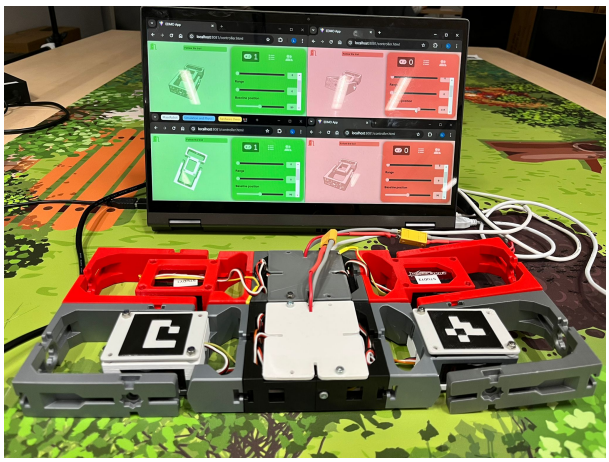


(b)

Figure 6.17: Cables' effect on the displacement of Snake 1 (a) and 2 (b)



(a) Initial position



(b) Matching position: Snake 1 (off1+25 and off2-1)

Figure 6.18: Difference in initial position between Snake 1 and 2

7 Limitations and Future Work

This study explored the performance and reliability of tracking and movement methods for the EDMOs, specifically Snake 1 and Snake 2. While the results provided valuable insights, several limitations were identified, which will guide future work.

7.1 Tracking Limitations

- Powering GoPro Cameras: The GoPros used for tracking do not support simultaneous charging and usage when operating on battery power. This limitation can be bypassed by removing the battery and powering the cameras through an USB-C cable connected to a power outlet or extension cord.
- Tracking Resolution and Noise: While 1080p resolution was employed for tracking, it still introduced noticeable noise, impacting the reliability of the measurements. Ideally, 4K resolution would improve accuracy, but the current setup faces significant delays in data transfer and processing, which limits real-time application. One potential solution is the use of GoPro's live streaming feature, which was briefly explored but not fully implemented due to time constraints. Further investigation into this feature is needed to determine its viability for real-time tracking.

7.2 EDMO Limitations

- External Factors: As discussed, external factors, such as cables, had a significant impact on the EDMOs' movement, introducing noise into the results. To mitigate this, we propose replacing cables with wireless communication systems, such as a WiFi module for data transmission and battery-powered motors. However, this approach requires further investigation to assess potential issues like WiFi latency, data loss, and the weight of the battery affecting motor performance.
- Motor Performance: Another significant limitation was the motor performance when transferring gait parameters from one EDMO to another. Differences in servo motor calibration between Snake 1 and Snake 2 made it difficult to directly transfer gaits. The motors also exhibited issues such as overheating, which reduced their torque and affected performance. To address these challenges, future work should focus on using more powerful motors or motors equipped with torque sensors to monitor and prevent overheating, thereby improving the consistency of movement.

7.3 Guiding children

To effectively guide children in optimizing gait parameters, we can leverage data from the fitness landscape and their exploration records by matching their current parameter set to the closest existing exploration path. This allows us to provide targeted suggestions that steer them toward a nearby local maximum, enabling

them to observe noticeable improvements in EDMO's movement. However, a quantitative threshold for what constitutes a perceivable improvement—one that children can both recognize and attribute to specific parameter changes—has yet to be established.

Ideally, such guidance would be personalized by tracking each child's exploration history to suggest the most effective next step. Although this parameter history is already available within the current framework, the optimal strategy for translating it into actionable and understandable guidance remains an open question. Due to time constraints, this aspect was not explored further in the current study.

8 Conclusion

Despite the limitations, this study successfully evaluated the tracking system using GoPro cameras and ArUco markers for motion tracking and explored the fitness landscape of the EDMOs. The experiments demonstrated the importance of GoPro settings in minimizing tracking errors, with 4K resolution and HyperSmooth offering the best accuracy, although at a higher computational cost. External factors such as cables were found to significantly affect the EDMOs' performance, underscoring the need for a stable, wireless setup for future work.

The fitness landscape exploration, guided by Powell's method, successfully identified optimal gaits for the EDMOs. However, transferring these optimal gait parameters between different EDMOs proved challenging due to discrepancies in motor calibration and servo performance. This highlights the need for further refinement in motor calibration and the use of consistent hardware across different EDMOs.

In the context of guiding children, the study's results suggest that by leveraging the fitness landscape and exploration data, it is possible to create an interactive system that provides personalized feedback to help children achieve optimal gaits. This approach could foster an engaging and educational experience.

Moving forward, future work should focus on improving the tracking system by exploring live streaming with GoPro cameras, addressing motor calibration issues, and refining the parameter transfer between EDMOs. By overcoming these limitations, we can enhance the reliability and functionality of the EDMOs, making them more effective for educational purposes and future research.

Bibliography

[1] Ashraf Alam. Employing adaptive learning and intelligent tutoring robots for virtual classrooms and smart campuses: Reforming education in the age of artificial intelligence. In Rabindra Nath Shaw, Sanjoy Das, Vincenzo Piuri, and Monica Bianchini, editors, *Advanced Computing and Intelligent Technologies*, pages 395–406, Singapore, 2022. Springer Nature Singapore.

[2] Sirajul Anwar, Nausheen A. Bascou, Muhsin Menekse, and Ali Kardgar. A systematic review of studies on educational robotics. *Journal of Pre-College Engineering Education Research (J-PEER)*, 9(2):Article 2, 2019.

[3] Larissa Barbosa, Sávio Freire, Rita S.P. Maciel, Manoel Mendonça, Marcos Kalinowski, Zadia Codabux, and Rodrigo Spínola. Attributes of a great requirements engineer. *Journal of Systems and Software*, 219:112200, 2025.

[4] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., Sebastopol, CA, 1st edition, 2008.

[5] Avraam Chatzopoulos, Michail Papoutsidakis, Michail Kalogiannakis, and Sarantos Psycharis. Action research implementation in developing an open source and low cost robotic platform for stem education. *International Journal of Computer Applications*, 178(24):33–43, 2019.

[6] F. K. Chiang, Y. Zhang, D. Zhu, X. Shang, and Z. Jiang. The influence of online stem education camps on students' self-efficacy, computational thinking, and task value. *Journal of Science Education and Technology*, 31(4):461–472, 2022.

[7] Martyn Cooper, David A. Keating, William S. Harwin, and Kerstin Dautenhahn. Robots in the classroom - tools for accessible education. In Christian Bühler and Harry Knops, editors, *Assistive Technology on the Threshold of the New Millennium*, volume 6 of *Assistive Technology Research Series*, pages 448–452, Düsseldorf, Germany, 1999. IOS Press.

[8] Linda Daniela and Miltiadis D. Lytras. Educational robotics for inclusive education. *Technology, Knowledge and Learning*, 24:1–4, 2019.

[9] GoPro. OpenGoPro. `https://github.com/gopro/OpenGoPro`.

[10] GoPro. OpenGoPro. `https://gopro.github.io/OpenGoPro/http`.

[11] GoPro. OpenGoPro. `https://gopro.github.io/OpenGoPro/tutorials`.

[12] Syahrul Bagas Himawan, Cucuk Wawan Budiyanto, and Aris Budianto. Technology self-efficacy in robotic-based integrated stem learning: A systematic review. In *Proceedings of the 6th Vocational Education International Conference (VEIC 2024)*, pages 135–142. Atlantis Press, 2024.

[13] Auke Jan Ijspeert. Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks*, 21(4):642–653, 2008. Robotics and Neuroscience.

[14] Sabina Jeschke, Akiko Kato, and Lars Knipping. The engineers of tomorrow: Teaching robotics to primary school children. In *Proceedings of the SEFI 36th Annual Conference*, Aalborg, Denmark, 2008.

[15] Sonia D. Jordan. Educational robotics and computational thinking in elementary school students. Electronic Thesis, 2023. Available at `https://digitalcommons.acu.edu/etd/725`.

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[17] Georgios Lampropoulos. Social robots in education: Current trends and future perspectives. *Information*, 16(1), 2025.

[18] MathWorks. Camera calibration. `https://www.mathworks.com/help/vision/ug/camera-calibration.html`.

[19] Peter Ngugi Mwangi, Christopher Maina Muriithi, and Peace Byrne Agufana. Exploring the benefits of educational robots in stem learning: A systematic review. *International Journal of Education and Development using Information and Communication Technology (IJEDICT)*, 17(1):91–115, 2021.

[20] Duyen Q. Nguyen. The essential skills and attributes of an engineer: A comparative study of academics, industry personnel and engineering students. *Global Journal of Engineering Education*, 5(1):65–75, 2001.

[21] OpenCV. Aruco marker detection. `https://docs.opencv.org/3.4/d9/d6d/tutorial_table_of_content_aruco.html`.

[22] OpenCV. Camera calibration. `https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html`.

[23] OpenCV. Example of markers images. `https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html`.

[24] Stamatios Papadakis. Robots and robotics kits for early childhood and first school age. *International Journal of Interactive Mobile Technologies (iJIM)*, 14(18):34–56, 2020.

[25] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992.

[26] Anton Sokolchenko SowmiyaNarayanan G. ArUCo-Markers-Pose-Estimation-Generation-Python. `https://github.com/GSNCodes/ArUCo-Markers-Pose-Estimation-Generation-Python`.

[27] Alexander Sproewitz, Rico Moeckel, Jérôme Maye, and Auke Jan Ijspeert. Learning to move in modular robots using central pattern generators and online optimization. *The International Journal of Robotics Research*, 27(3-4):423–443, 2008.

[28] TeamEDMO. `https://github.com/TeamEDMO`.

[29] Hannah Thinyane. Academic perceptions of the ideal computer science student. *Computer Science Education*, 20(3):229–247, 2010.

[30] Lai Poh Emily Toh, Albert Causo, Pei-Wen Tzuo, I-Ming Chen, and Song Huat Yeo. A review on the use of robots in education and young children. *Journal of Educational Technology & Society*, 19(2):148–163, 2016.

[31] Maastricht University. Edmo - modular robotic platform. `https://www.maastrichtuniversity.nl/edmo`.

[32] Wikipedia. Pinhole camera model. `https://en.wikipedia.org/wiki/Pinhole_camera_model`.

[33] Wikipedia. Stochastic gradient descent . `https://en.wikipedia.org/wiki/Stochastic_gradient_descent`.

[34] Douglas C. Williams, Yuxin Ma, Louise Prejean, Mary Jane Ford, and Guolin Lai and. Acquisition of physics content knowledge and scientific inquiry skills in a robotics summer camp. *Journal of Research on Technology in Education*, 40(2):201–216, 2007.

[35] Jinyu Yang and Bo Zhang. Artificial intelligence in intelligent tutoring robots: A systematic review and design guidelines. *Applied Sciences*, 9(10), 2019.

Appendix

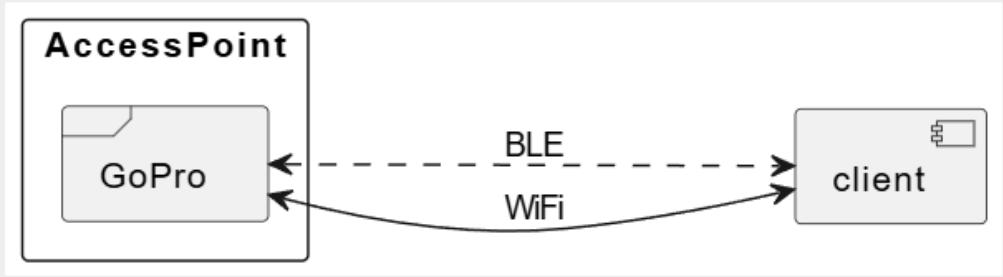


Figure 8.1: GoPro connection via wifi [11]

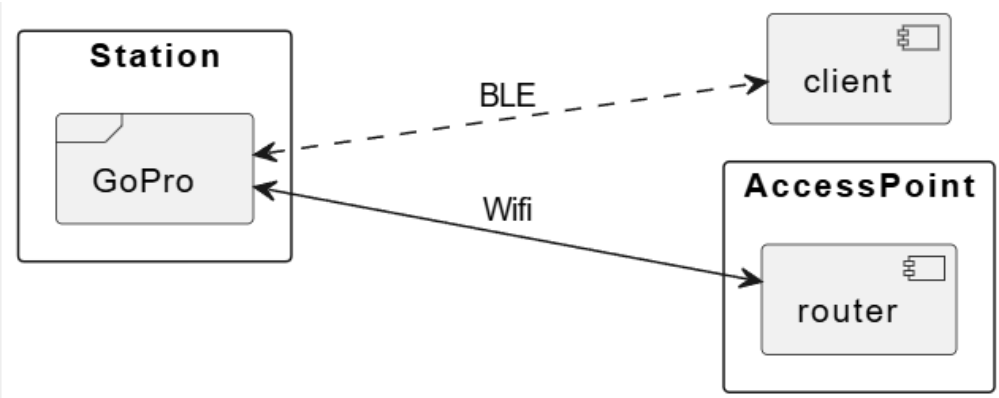


Figure 8.2: GoPro connection via COHN [11]

Command	Function
`camera control`	Get control of the camera; the camera will return to the idle screen.
`set resolution 1080`	Set the resolution to 1080p.
`set resolution 4K`	Set the resolution to 4K.
`set resolution 5.3K`	Set the resolution to 5.3K.
`set video`	Set the camera mode to video.
`set photo`	Set the camera mode to photo.
`get last video`	Download the last video in the current directory.
`get video`	Get a video by specifying the video's folder and file name.
`start`	Start a video in video mode or capture a photo in photo mode.
`stop`	Stop the video if recording; otherwise, do nothing.
`get preset`	Get the available presets.
`load preset`	Load the preset by ID (defaults to 6).
`keep alive`	Send every 3 seconds or change camera sleep time.
`start stream`	Start a stream.
`stop stream`	Stop the stream.
`get camera state`	Get all camera settings and statuses.
`get camera battery`	Get the camera's battery level (0-3, 4=charging).
`get cohn state`	Get COHN status.
`get media list`	Retrieve the media list. See documentation.
`turbo mode`	Enable faster downloads via Wi-Fi.
`hilight`	Add a highlight to the GoPro.

Table 8.1: A selection of available commands to control the GoPro through the OpenGoPro HTTP API [10].

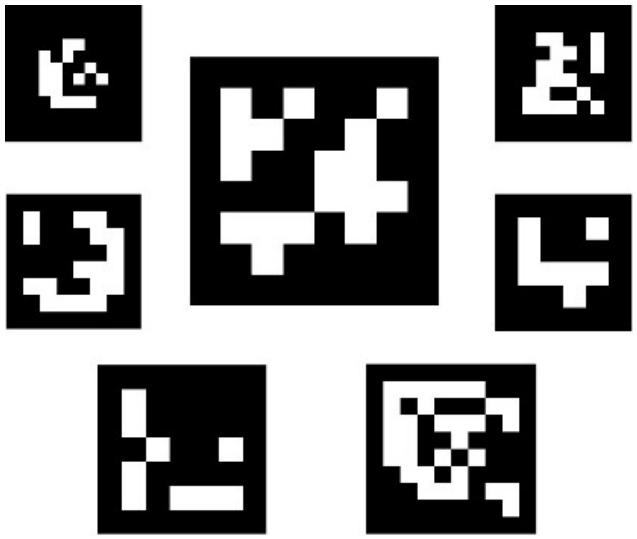


Figure 8.3: ArUco markers [23]

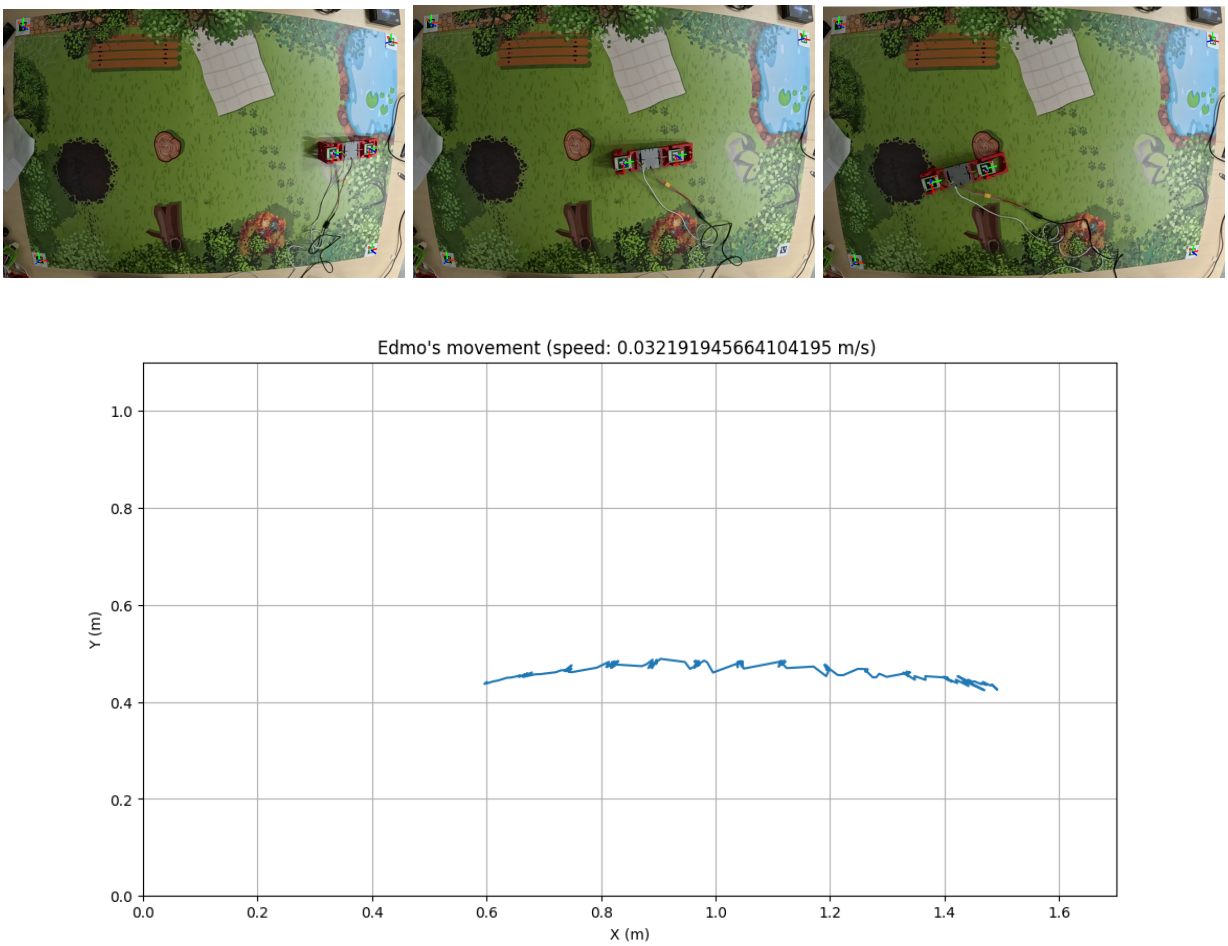


Figure 8.4: Example of ArUco marker detection and position estimation using the ArUco_Markers_Pose package.

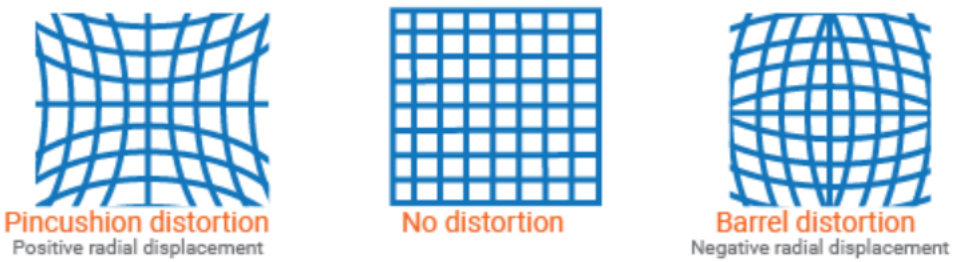


Figure 8.5: Radial distortion examples [18]

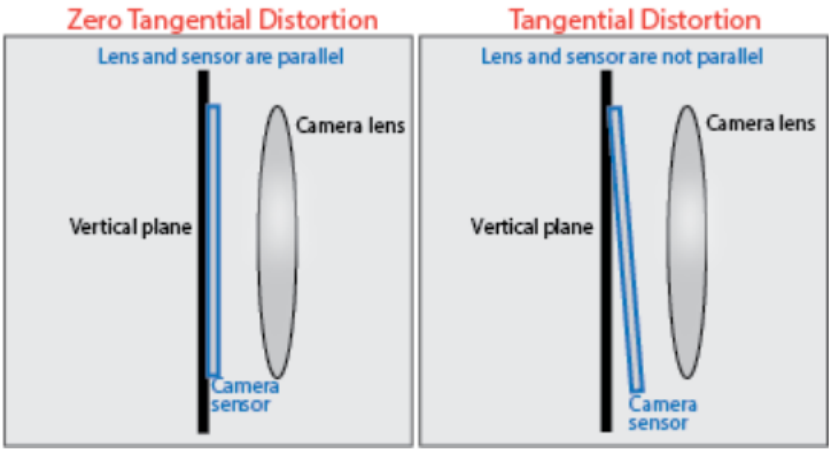


Figure 8.6: Tangential distortion examples [18]

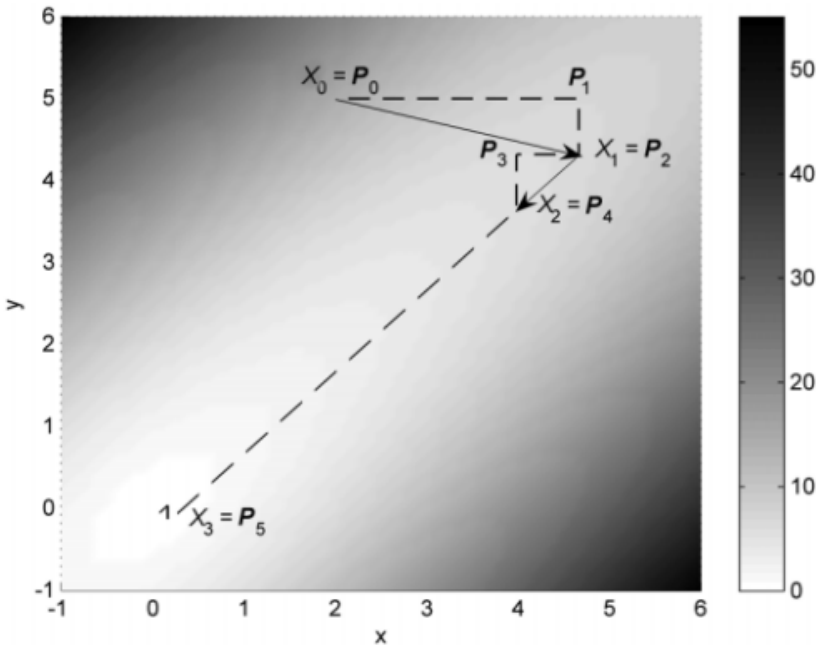
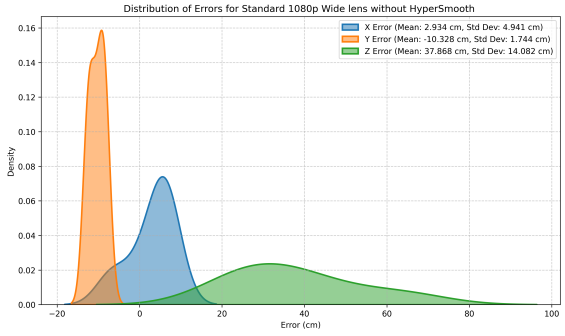
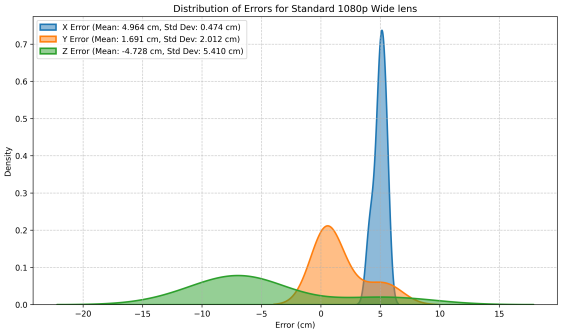


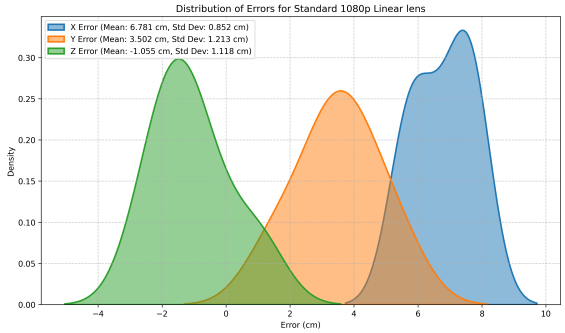
Figure 8.7: Illustration of powell's method: "Powell's method starts with the unit vectors $e_1, e_2, \ldots, e_N$ of the N-dimensional search space as a set of directions. One iteration of the algorithm completes N line minimizations along the N directions in the set. The algorithm is illustrated ... for the two-dimensional function ($f(x, y) = x^2 + y^2 + (x - y)^2$). Starting at the initial point $\mathbf{P_0} = (2, 5)$, the first line minimization along the direction given by the unit vector $[1, 0]^T$ takes us to the point $\mathbf{P_1}$. From this point the second line minimization along $[0, 1]^T$ takes us to $\mathbf{P_2}$ and completes the first iteration. As can be seen in the figure, repeated line minimizations along the unit vectors would involve many iterations because the minimum would be approached in small steps. After each iteration, Powell's method checks if it is beneficial to replace one of the directions in the set by $\nu_i = \mathbf{P_0} - \mathbf{P_N}$, where $\mathbf{P_0}$ was the starting point at the current iteration and P_N the new point after the N line minimizations. In the example figure, ν_2 replaces $[1, 0]^T$ in the second iteration. The algorithm correctly decides not to include new directions in all other iterations as this would actually slow down convergence. The mechanisms for deciding whether or not to include the new direction ν_i after each iteration and which direction in the set should be replaced are described in Brent (1973) and Press et al. (1994)[25]."-adapted from [27, p. 431].



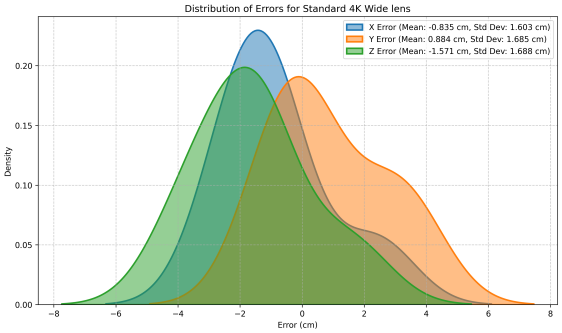
(a) Standard 1080p Wide lens without HyperSmooth



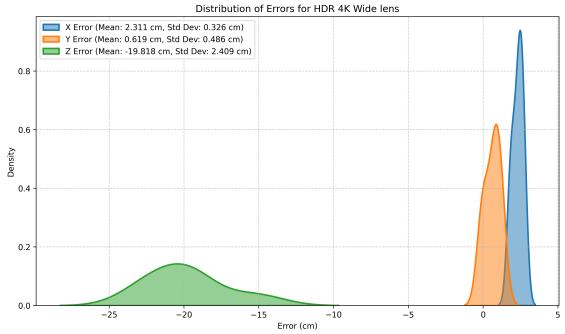
(b) Standard 1080p Wide lens



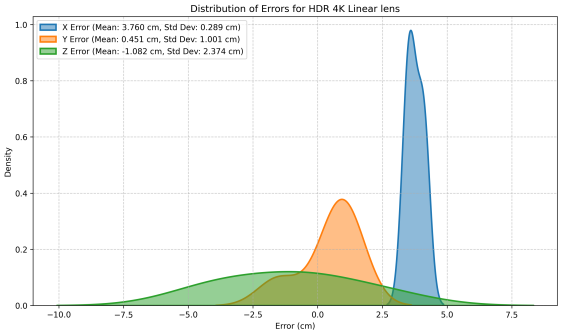
(c) Standard 1080p Linear lens



(d) Standard 4K Wide lens

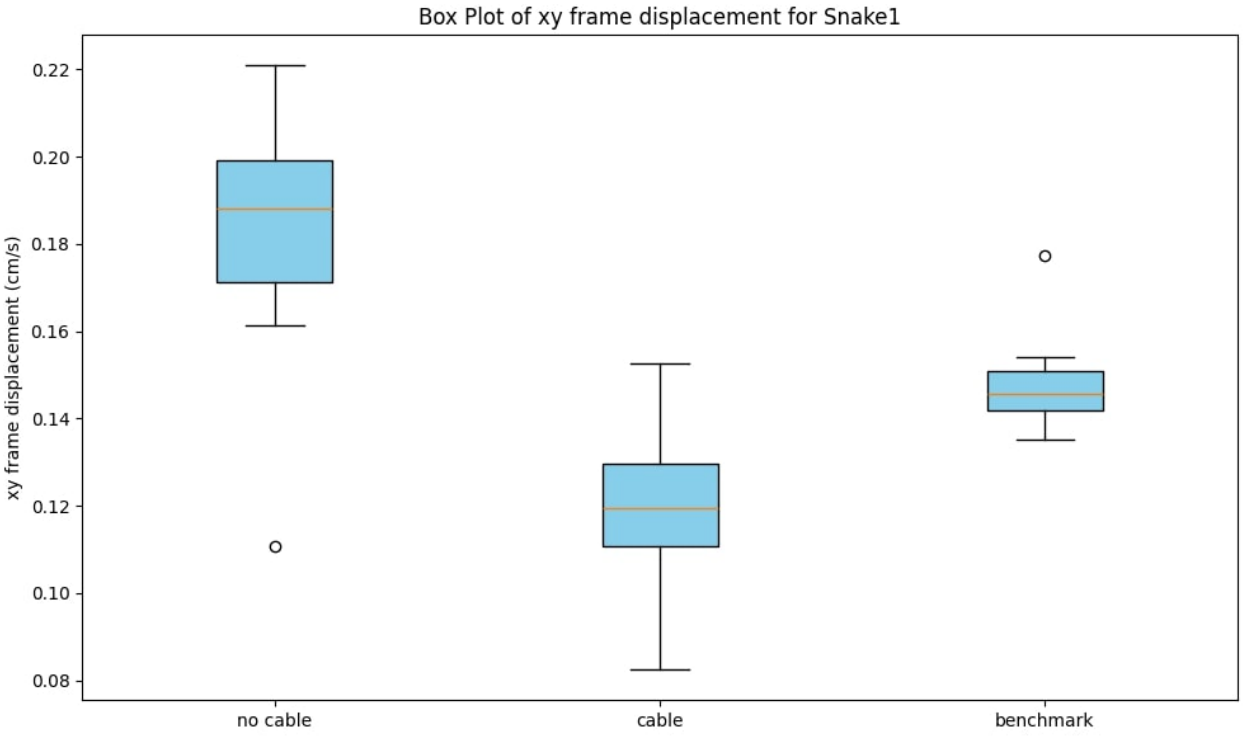


(e) HDR 4K Wide lens

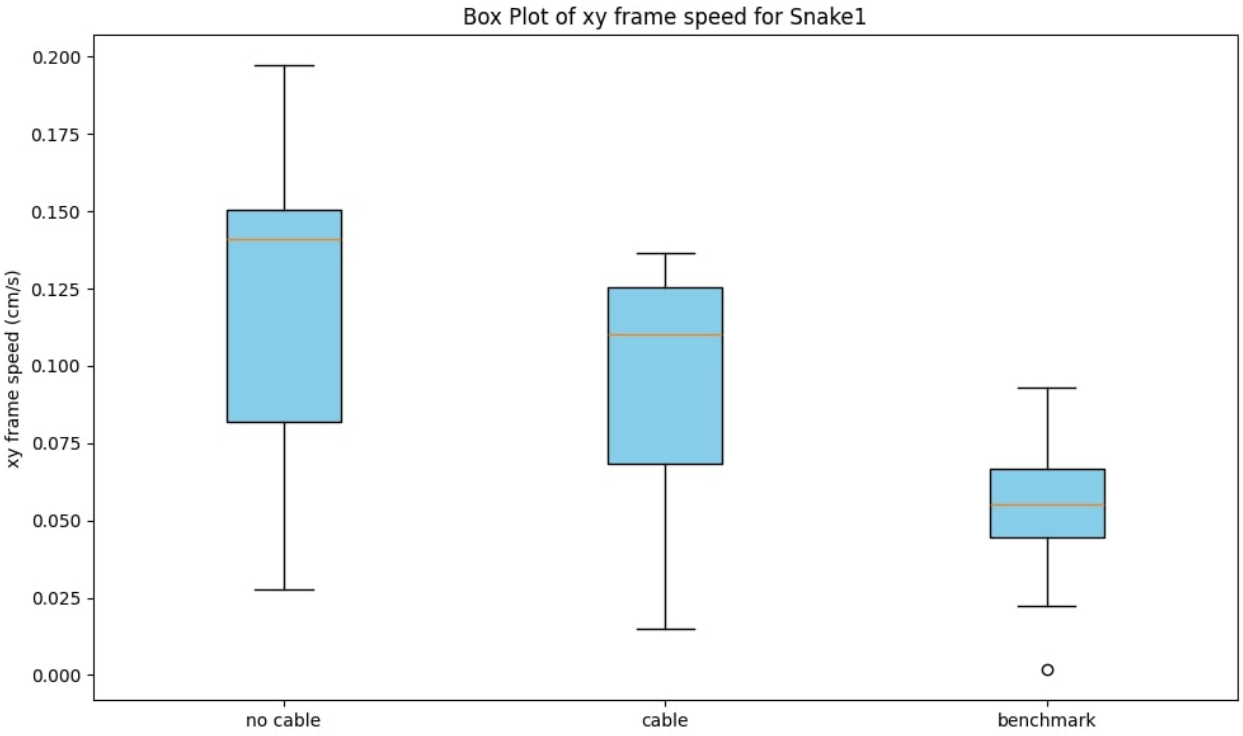


(f) HDR 4K Linear lens

Figure 8.8: Comparison of GoPro settings error distributions



(a) Speed computed from the distance traveled between the first and last captured frame.



(b) Speed computed from the average of distance traveled between successive frames

Figure 8.9: Both plots are based on the same experimental data.

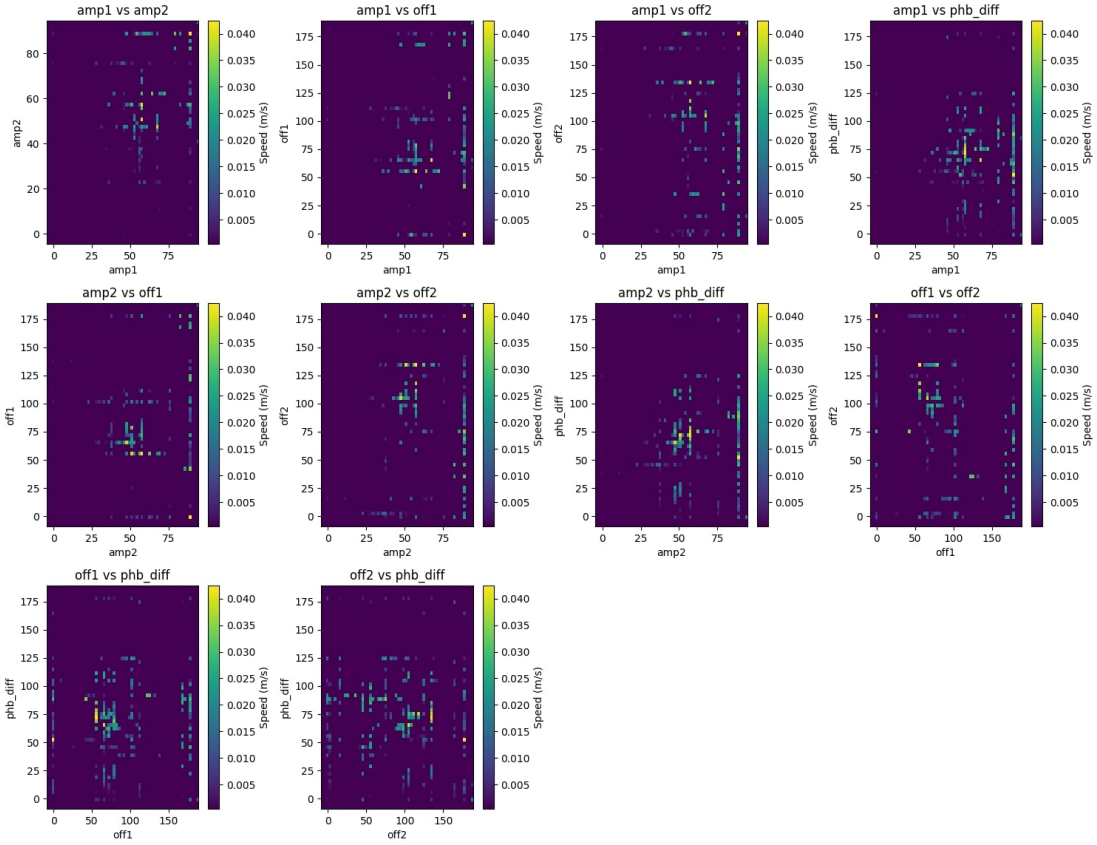


Figure 8.10: Fitness landscape for Snake 1

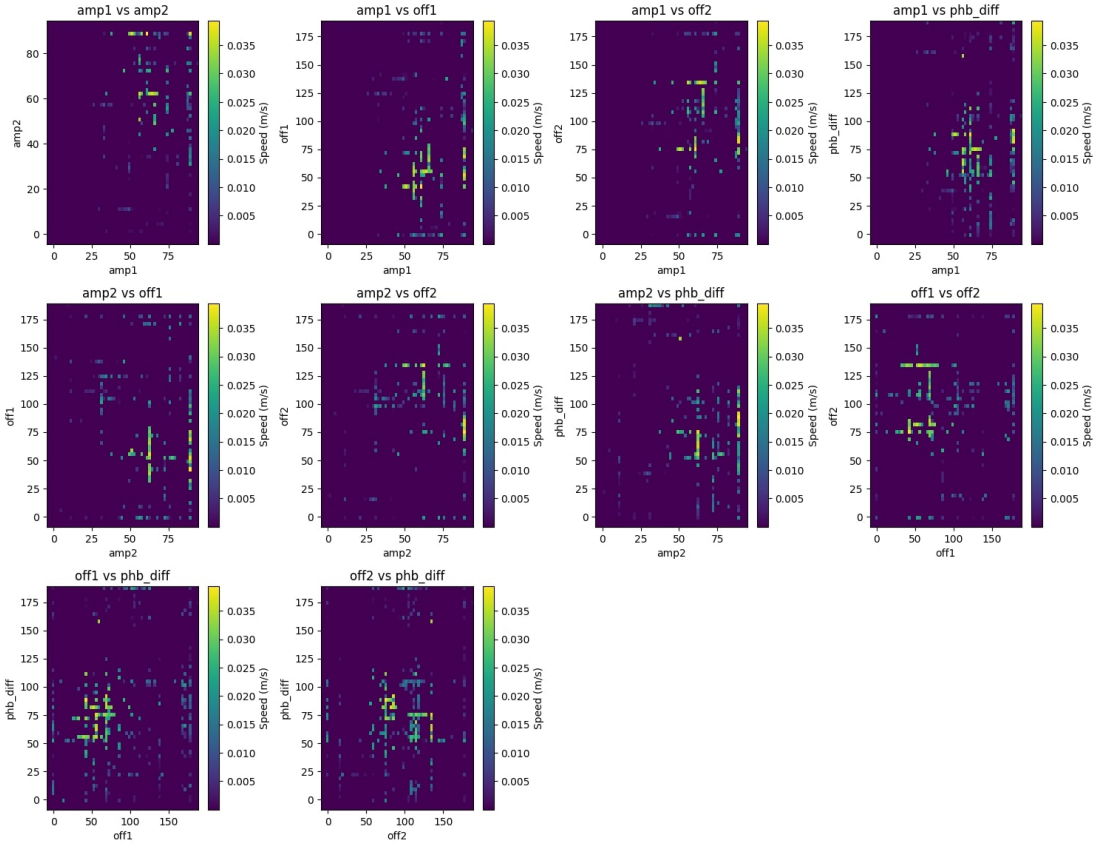


Figure 8.11: Fitness landscape for Snake 2

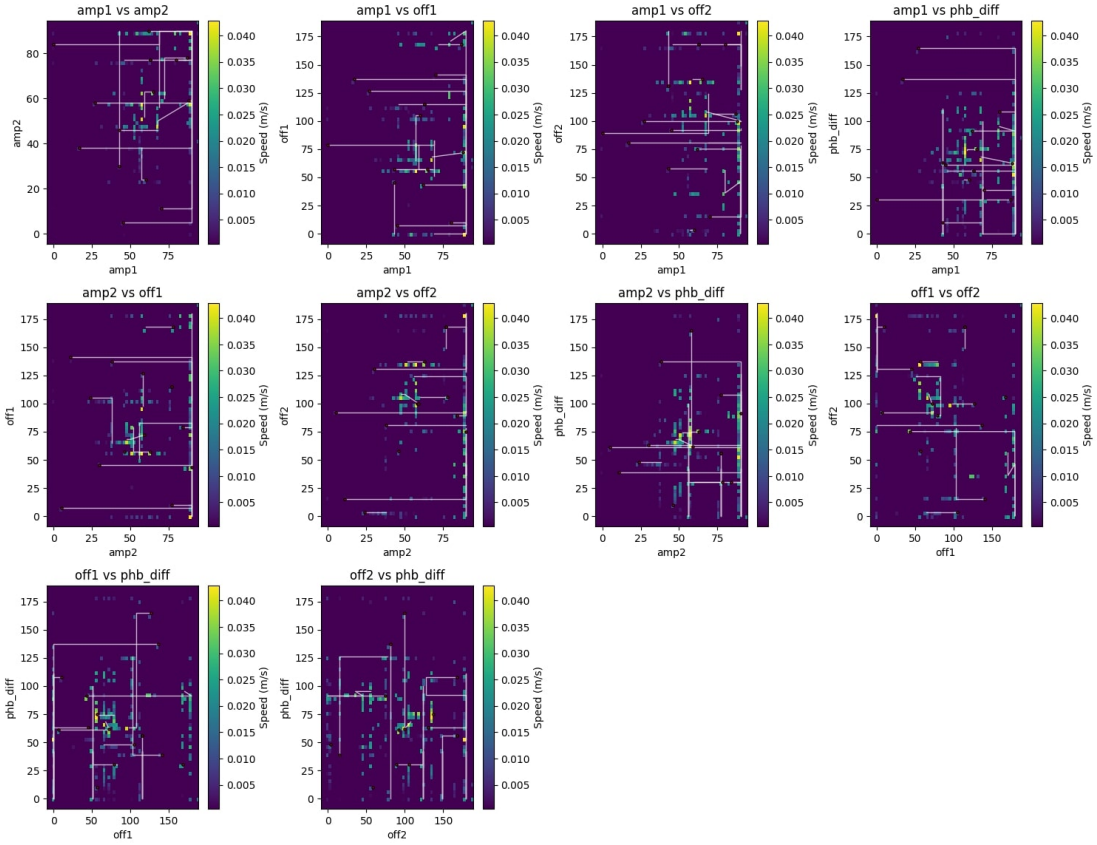


Figure 8.12: Exploration paths in the fitness landscape for Snake 1

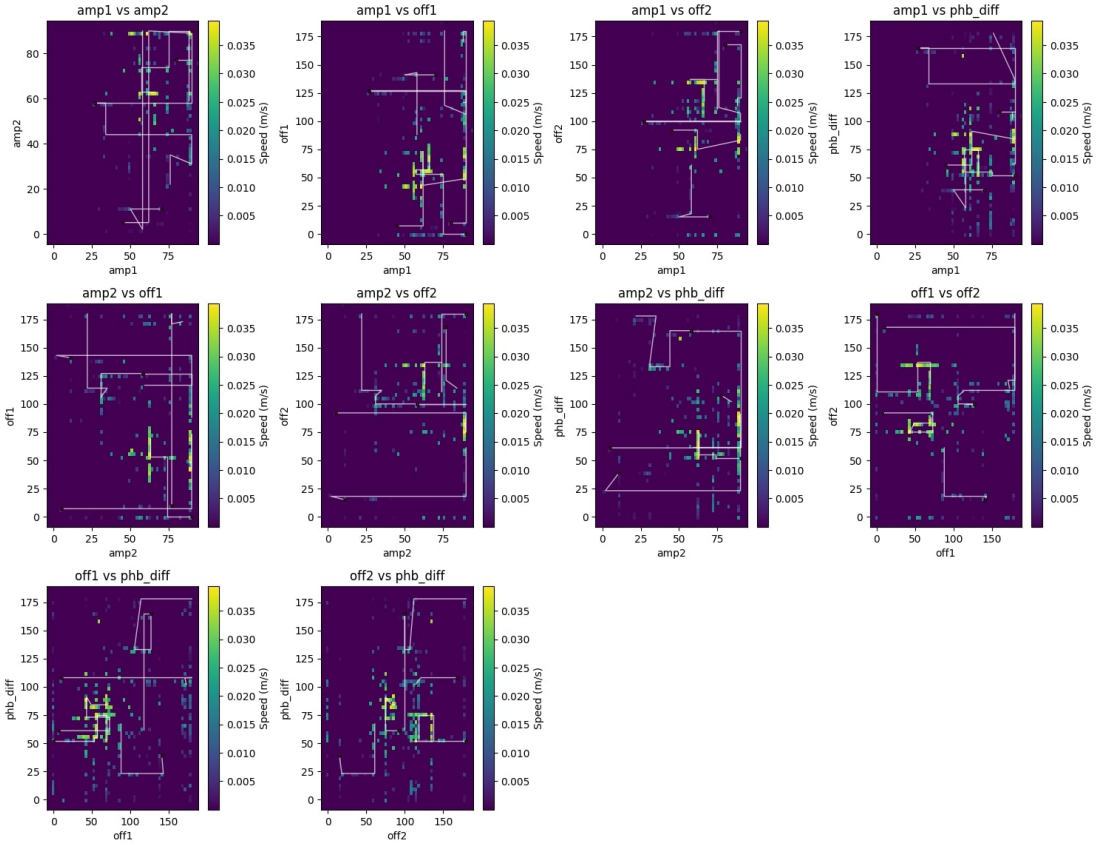
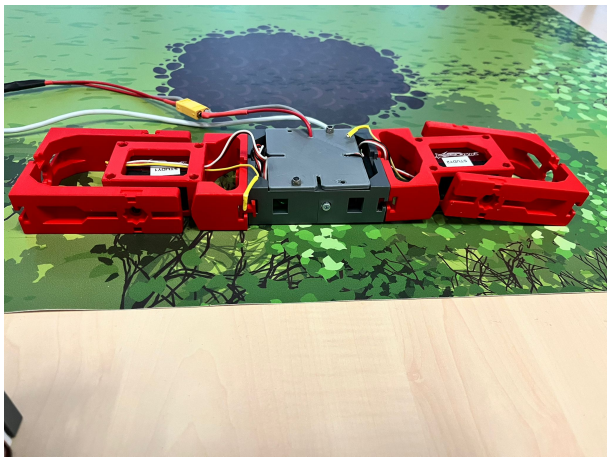
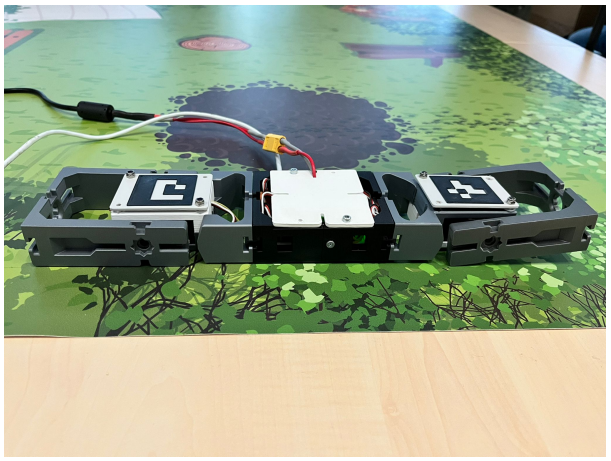


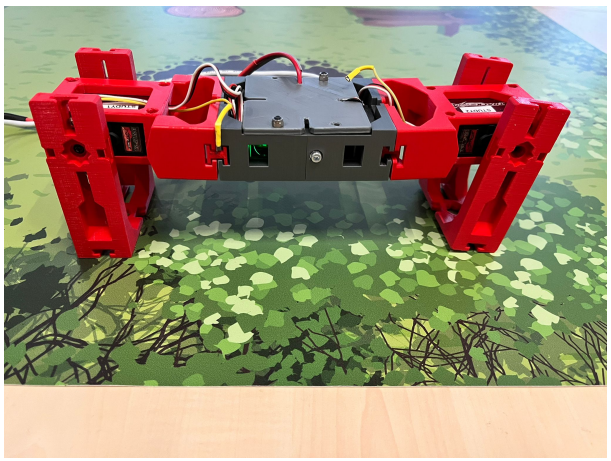
Figure 8.13: Exploration paths in the fitness landscape for Snake 2



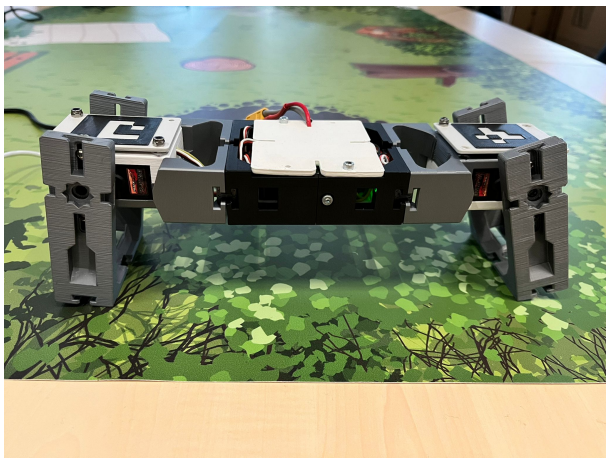
(a) Snake 1 initial position



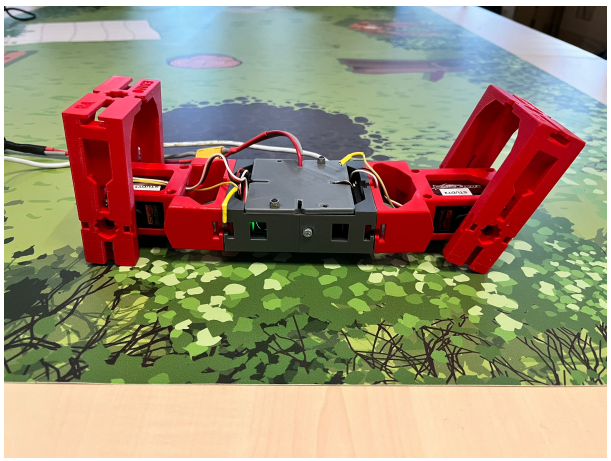
(b) Snake 2 initial position



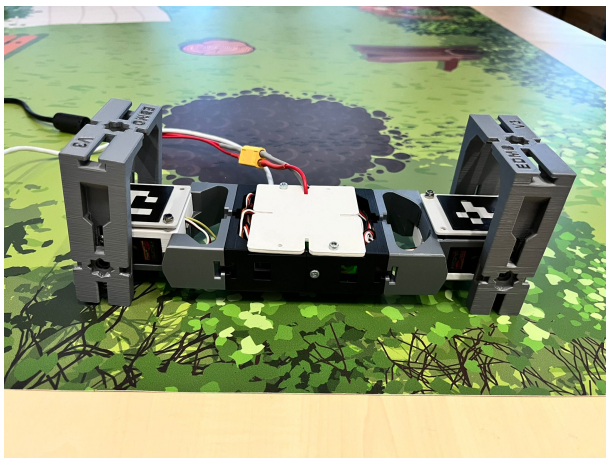
(c) Snake 1 bottom position



(d) Snake 2 bottom position



(e) Snake 1 upper position



(f) Snake 2 upper position

Figure 8.14: Difference between Snake 1 and 2's range of motion