Dynamic Recalibration of the Odometry System Through Markers to Improve Travel Precision

Yuanbang Liang I.D. 2047910
Marina Montalban de la Mata I.D. 2051552
Emanuel Nunez Sardinha I.D. 2073043
Keshava Shankar Nagu I.D. 2070991

***Abstract*—A measure of performance in odometry systems during autonomous exploration is the match between internal kinematics and its real-world location. In this experiment, we aim to improve this metric on a line map, measured as the deviation in X and Y axis and rotation angle with respect to a starting point. For this we propose a node system added to the maps acting as points of interest (POI), which the robot will use to recalibrate its kinematics. By returning to these POI bypassing the map, we show that error due to drift can be reduced in certain applications. To achieve this we conducted three sets of experiments: evaluating a refined exploration algorithm, testing the optimal distance for travel between nodes, and large scale testing. Kinematic recalibration was shown to work in test cases, with the average final positions 25% closer to truth, compared to identical routes without calibration.**

I. Introduction

A very simple and accurate localisation system for robots is odometry [1]: calculating the robot's displacement from effectors movement, wheel turns in the case of wheeled robots. Through trigonometry, position can be deduced from the movement of each wheel, calculating horizontal, vertical and final angle respect a reference point. The importance of a precise odometry system can be illustrated by numerous robots conducting autonomous exploration tasks in rescue or maintenance, where they are sent out on a track with a set objective [2], [3]. As a step for a high-quality self-location system, the overall performance and reliability of the exploration would significantly improve by ensuring a proper home return. However, it has marked limitations, namely drift produced by imperfect calibration and inaccurate representation of physical variables in software [4].

This drift only grows with exploration time, resulting in a mismatch between internal kinematic variables and the real-world values. Thus, precision lowers over time, and tasks dependant on accurate position become more difficult. This issue can be approached by adding reference markers to the map to explore [5]. The use of markers on tracks in explorations has many popular applications, particularly as they serve as location references throughout the track and can therefore contribute to improvements in an accurate movement.

For these reasons, we propose overlaying points of interest over a track, and use a device capable of recording their position. The position of each point $\vec{p_i}$ will be actually accompanied by an error due to drift $\vec{p_i} + \vec{\epsilon_i}$ We proceed under the assumption that, since error grows with time, points found later during exploration will be more affected by drift. Furthermore, we assert that the error caused by the recalibration process will be less than the cumulative navigation:

$$\sum_i \vec{\epsilon_i} < \vec{\epsilon_0} + \sum_i \vec{\epsilon}_{calibration\ i} \tag{1}$$

Thus, by returning to a previous point of interest, centering and calibrating kinematic values, is possible to reduce the error element in the kinematic system to that of an early point in time plus a small calibration error, $\vec{p_i} + \vec{\epsilon_j}$ where $j < i$ (Figure 1). This process can be then carried in succession over long distances. A trade-off arises when increasing the number of nodes and thus recalibrations, as the total recalibration error increases

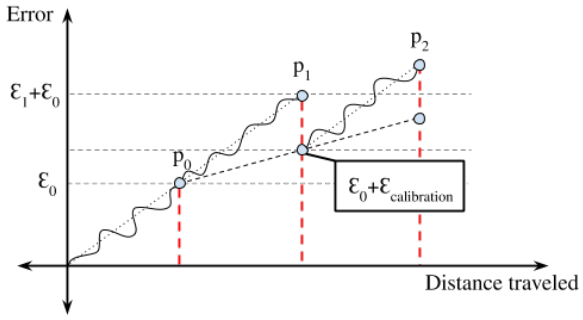


Fig. 1: Error reduction through recalibration

A similar procedure was studied by Paijens et al. [6], where computer mouse sensors are used to recalibrate a mobile robot when the floor surface changes. Other least-squares techniques for recalibration has been validated [7]. Using sensory data on wheel velocity, Ishigami et al. [8] shows a similar intention to improve odometry for the case of outdoor terrain.

For the experiment four Romi Pololu robot systems were used, incorporating an array of three light sensors. A set of maps was constructed in black tape over white paper, with features including curves, sharp turns and gaps. Following the line is the exploration task, with added complexity of returning to the start point once finished. The final position after returning will be used as the measure of the final error $\sum \vec{\epsilon_i}$. Nodes were placed at key points along the track, but due to the drift error their separation was experimentally determined.

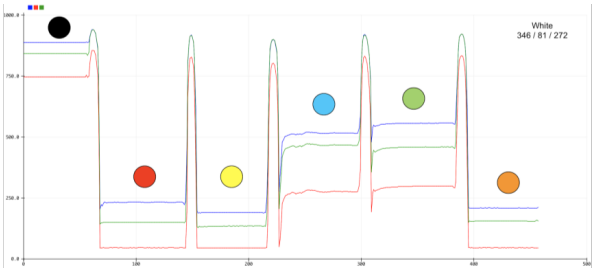
A. Hypothesis Statement

The simple task of returning to the starting point after conducting a line-search exploration of a map is a holistic measure of the performance of a robot's odometry system. The performance error increases with the exploration time and distance, drifting from its initial value. The return position and angle with respect to the initial conditions may vary at each repetition, but fall under a measurable distribution. By assigning reference points in the track and returning to them, it may be possible to reduce the travel error.

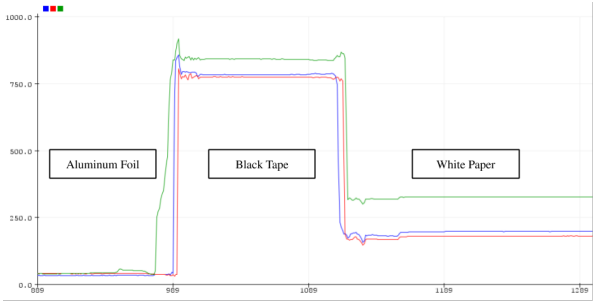
> *By overlaying a node system onto a explorable route and conducting a calibration routine at each node, the error can be bounded by a threshold determined by the distance travelled between calibrations. Optimal positioning of the nodes will provide a good trade-off between exploration time and return accuracy, evidenced by improved precision.*

We will compare performance on a simple and a modified map, measuring the difference in position after returning to a starting point. Given how the light sensor used is built specifically for the line following task, we incorporate a second supplementary hypothesis:

> *The light sensor can be used to clearly distinguish between surface, track and points of interest, used to construct the overlaying nodes. An optimal material can be found to construct nodes with.*



(a) Light sensor response for tape and different paper colors



(b) Light sensor response for black tape, aluminum foil and paper

Fig. 2: Light sensor response for different materials

II. Implementation

A. Initial Setup

For the implementation, a preexisting line-follower code for the Romi was used as starting point, with additional features from Dr. O'Dowd version [9]. This algorithm features ability to follow lines marked in tape at sharp angles, cross small gaps and perform some simple exploration. It also features an interrupt-activated odometry system, using the encoders to track the robot position as it moves through simple kinematics. Kinematic values and speed are updated at 50Hz.

The implementation used a Pololu QTR-3A Reflectance Sensor Array light sensor [10], capable of detecting dark markings via IR reflection. Though intended for line following, the sensor outputs 3 analogue voltages corresponding to light intensity, opening up the door to use it for our node application. To assess the viability of this experiment, an initial response test was carried out to select a suitable node material. The reflectance of different candidate materials was measured, including a variety of papers, cardboard, and several reflective materials, as shown in Figure 2. Among other factors, larger differences in readings suggest a better node material. Aluminum foil was selected due to its high reflectance, resulting in a noticeable better performance due to the high contrast with opaque black tape, as seen in Figure 2b.

B. Nodes

The nodes are constructed from aluminum foil, cutting 2 cm diameter circles and overlaying them onto the tracks, as shown in Figure 3, matching the reflectance sensor coverage. The existing line recognition code was expanded to accommodate the nodes, based on a threshold. A *node confidence* parameter is calculated based on the number of sensors detecting a node at 50Hz and processed with a low-pass filter. This value reaches its peak when the sensor is centered within two milimeters of a node. A calibration procedure occurs at startup, where the Romi assigns the value of the white paper as zero. Upper and lower threshold values are used to distinguish whether the Romi is on a path or a node.

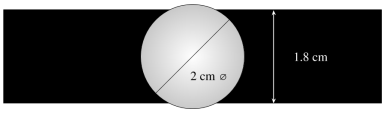


Fig. 3: Node design over the track

In order to properly calibrate the robot based on the node position, additional considerations were taken. The baseline performance of the kinematics in the existing code was overhauled, further explained in Section II-C. During the calibration procedure, a small trigonometric procedure is performed to offset the sensor position, since it is not aligned with the robot center of motion, as shown in Figure 4.

Internally, nodes are modelled by a *node class* and managed by a *node manager class*, which features several helper functions, including a distance check to prevent repeatedly adding the same node. This system also allows the Romi to access information on previous nodes when found - such as position with respect to the beginning and rotation of the Romi when encountered - and finds routes between them. The sensor

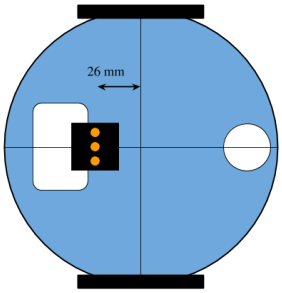


Fig. 4: Sensor placement on the Romi

manager class was expanded to account for the material of the black tape path and the aluminum nodes. This allows the Romi to recognise a node with minimum effect over the navigation process, and applies a low-pass filter to the resulting signal. The main state machine was expanded to account for moving back and forth between nodes, centering in nodes, recalibrating and flow of control.

Following that, new features were incorporated to our program to allow the Romi to conduct regular calibration routines throughout the exploration. These include node recognition and detection, storing node locations, finding shortest paths through the odometry system and continuing the exploration task. These were tested first independently of the targeted map to ensure correctness.

C. Accuracy Improvements

In order to guarantee accurate movement between nodes, the accumulated kinematic error had to be bounded to similar to the diameter of the individual node. Three key improvements on the software made this possible[1]:

- The internal constants for conversion between distances and encoder ticks for rotation and translation where re-adjusted separately and tested experimentally over large surfaces. Wheel radius and turn radius was determined experimentally per Romi.
- Previously, the underlying kinematic system used two `long` type variables to store X and Y displacement. This data type is ideal for storing large numbers, such as the displacement in encoder counts. However, it has a significant limitations as it can only store integer values, truncating decimals and discarding information of fractions of a movement. Other data types allow decimal data, but lose precision when storing large values. The current kinematic system features a *mixed data type* approach, with the decimal part stored as a `float` and the integer as a `long`.
- The existing function to move to specific coordinates was revised. It works as a simple "point and go", aiming in the direction given by coordinates and moving the required distance while updating kinematics, but is executed multiple times when traversing long distances, significantly reducing the effects of imperfect start orientation.

[1]Final code can be found on https://github.com/tinyAtlas/Dynamic-Recalibration-of-the-Odometry

D. Algorithm

Figure 5 shows the implemented calibration routine. During normal operation, the *node confidence* parameter is updated and pooled, and if it increases past a threshold, the robot performs the calibration procedure. If there is more than one node in memory, it goes to the penultimate's position - ignoring the track - and performs a short exploration routine to find its center. There it looks for the highest *confidence value* and its pairing coordinates. If successfully positioned, the Romi updates its internal kinematic values and those of the latest node. The Romi then returns to the latest node position, and repeats the procedure at each subsequent node.

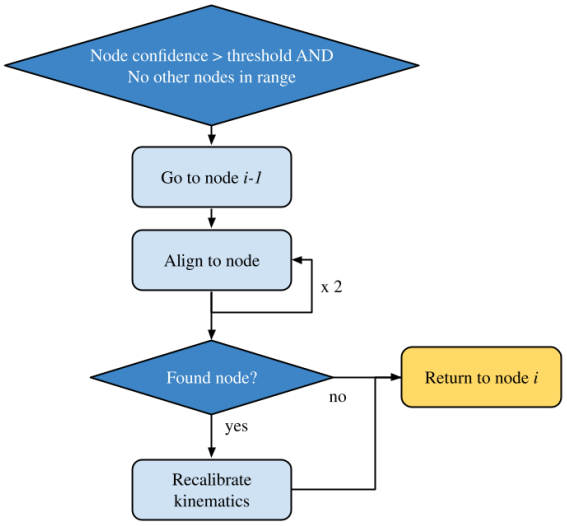


Fig. 5: Overview of the calibration procedure

For the multi-stage centering routine, we perform a search very close to the return position. Exploiting the fact that the light sensor is slightly offseted from the center, the Romi:

1) Rotates in place in both directions, recording the angle with the highest node confidence value.
2) When finished, it points in the direction of maximum confidence.
3) The robot moves back and forward in place, again looking for the point of maximum confidence and its coordinates.
4) The process repeats, by default twice.

If successful, the Romi has in memory the node with maximum confidence, event if not physically over it. Small motion kinematics are updated during the course of the exploration.

III. Experiment Methodology

A. Overview of Method

The experiment is divided into three main phases: benchmarking the base performance when returning from exploration, determining the optimal separation for the nodes, and

repeating the benchmark process for the finalised algorithm, comparing as per the metrics defined in section III-C.

1) Experiment 1: To assess the base performance of the odometry system, the improved base algorithm is tested on a variety of maps, and the returning accuracy and elapsed time are recorded. Accuracy is obtained from the distance from the a common origin, set as the starting point for all tests, marked on the map. The return position in X and Y, and angle θ are defined as shown in Figure 6.

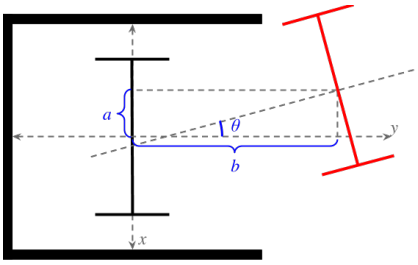


Fig. 6: Measurement convention for X, Y and θ

2) Experiment 2: In order to control for the main independent variable, the node spacing, we study its small scale effects. For four representative track segments, we experiment with different separations between the nodes, determining optimal distances and using the results to inform node placement on the last section. The track was divided into full maps and isolated sections formed by straight lines, U-curves and 90° to both sides, as shown in Figure 7. We seek for the maximum distance between nodes that allows for consistent node localization, and as such, we will repeatedly tests for this instances.

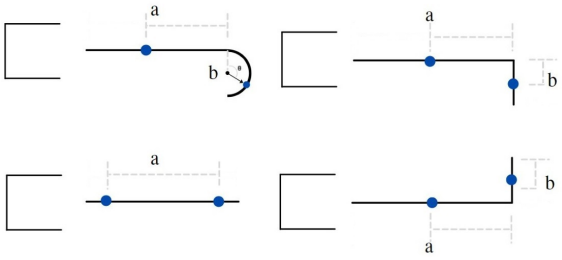


Fig. 7: Test segments, clockwise from top: Right "L", Left "L", Straight line, Curve. Segments not to scale

3) Experiment 3: Finally we will replicate the initial tests on the modified map, now overlaid with a node system. We will collect the performance data and analyse it against the initial experiment. In this way we will quantify the performance gain through a regular re calibration routine in a map exploration.

B. Discussion of Variables

- **Controlled Variables**: In order to have a good representation of the general performance, the modularised sections of the tracks will be studied, followed by a selection of different courses. When returning to a previous node for calibration, there is a chance the error has grown large enough for the robot to fail returning. If the recalibration process fails, the robot simply returns without updating position, and sends an audio cue. These instances will be noted during testing.
- **Independent Variable**: Node spacing, i.e. the distance between successive nodes, is the main independent variable through the testing process. Placement of the nodes in the final tracks is informed by Experiment 2, testing different separations in individual track segments.
- **Dependent Variable(s)**: The precision and accuracy when returning home are the core variables of interest, defined as the difference between initial and final position in x and y, and rotation angle. We will refer to this throughout the report as the performance or precision error.

C. Discussion of Metric(s)

For this experiment, we defined metrics of error and performance for accuracy in conducting a map exploration following a non-reflective tape path and returning to the origin. The return position is marked and measurements are taken on the centre's (i.e. the midpoint of the two wheels) side deviation, height and rotation angle with respect to its starting position.

We assessed the level of error based on an average of first results -on track segments and scaling on maps- taking into account best performance cases and improvement capability. Throughout the testing, we also took into account the precision against the accuracy. A high precision with low accuracy suggest a consistent level and direction of error. These are reflected in the average and standard deviations described in Section IV.

IV. Results

A. Experiment 1: Baseline performance

Comparison between the baseline accuracy of the algorithm appears in Figure 8, showing the mean and standard deviations of the return position and angle. We can observe vast improvement over a long exploration period, with the mean position deviation decreasing from 11.7cm to 3.9cm with a similar deviation. Average return angle is 2.6 degrees of the center position, but with a wider spread.

The error was bounded to a similar scale as a node's width, which thanks to the exploration routine, should allow the Romi to successfully reach subsequent nodes. However, the error remains a function of the distance traveled, which requires further study to control for node separation.

B. Experiment 2: Node Separation

Results for node detection appear in Figure 9. X axis shows the separation between the nodes *along* the track, and Y axis the percentage of successful returns. The cutoff point used has been marked for each.

During the tests, the Romi reached previous nodes consistently to a surprisingly separation. Sharp right turn segments were reachable up to a combined distance of 60cm, while left

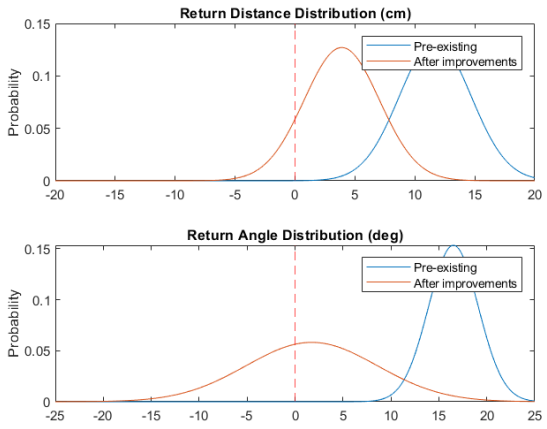


Fig. 8: Distribution for baseline vs improved performance during complete map exploration (60 and 30 tests respectively on 6 circuits)

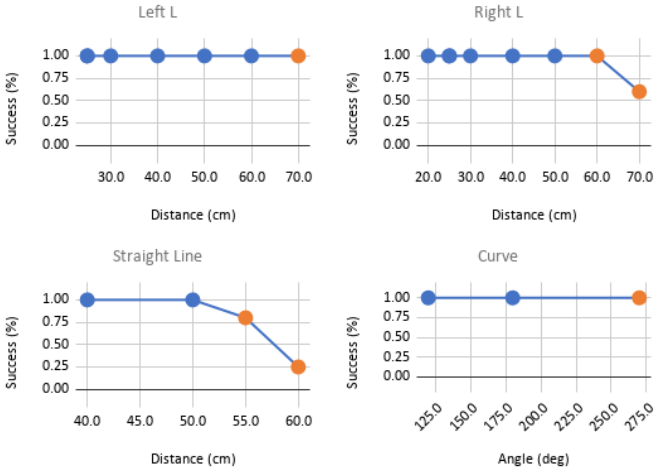


Fig. 9: Cutoff values for distinct tracks as a function of distance (40 tests each)

turns segments exceeded the planned test space, caping both at 60cm. Straight segments remain consistently reachable up to 50 cm. Finally, sharp curves (15 cm radius) remained fully reachable for all tested angles. Experiments were conducted in several Romis, and irregular performance was observed in some models, possibly due to slightly different sensor capabilities.

This phase of testing suggested that the algorithm can be used with confidence to detect nodes and conduct recalibration and travelling tasks. We can assume few restrictions when placing nodes, aside from a approximate max distance 60 cm. The three improvements have bounded the error to a 2 cm point for moving between close points (less than 50cm apart), with almost negligible difference in start and final angle (80 tests, 2 outliers discarded). With this knowledge we can test the large-scale algorithm.

C. Experiment 3: Algorithm Comparison

The final recalibration algorithm was directly compared with the initial code, implementing all precision improvements but ignoring all nodes heading straight to the goal. Results for 3 circuits are shown in Figure 10.

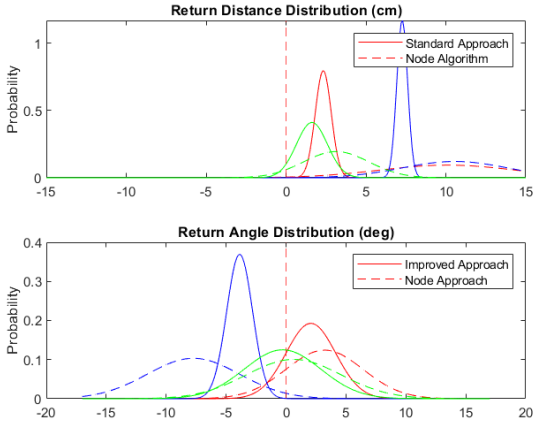


Fig. 10: Performance comparison of improved approach (30 tests) and node approach (50 tests). Different color pairs refer to different circuits

On a one to one comparison, the recalibration algorithm proves inferior to the traditional approach. in the final position varying from more than 8 cm and angle from $\pm 10°$ (60 tests in total). The improved "normal" algorithm displayed outstanding consistency at arrival, and very small error in two of the circuits tested, with less than 2 cm from the starting point and angle within 5 degrees. Additionally, a recurrent issue was encountered with inconsistent behaviour in some Romis, failing to distinguish white paper from aluminum, hampering the number of tests.

D. Effect of calibration

Given the previous results, it is unclear if the kinematics recalibration has an actual effect on final accuracy, as the node approach moves considerably more during execution, with an exploration time and distance roughly three times longer than before. To address this, we prepared a small scale experiment comparing performance for identical movement. The node algorithm was compared against itself without the recalibration feature, meaning the Romi move back to previous nodes and centers in place, but does not recalibrate its kinematics. Results appear in Figure 11.

Results suggest a noticeable improvement with a drawback. After calibration the average for the displacement is lower but standard deviation grows, meaning that on average the Romi returns to a point closer to the start, but over a wider area. This points to an increase in accuracy (overall result) at the cost of precision (consistency).

V. Discussion and Conclusion

The aim of this experiment was to improve accuracy of a Romi conducting a map exploration through searching a line

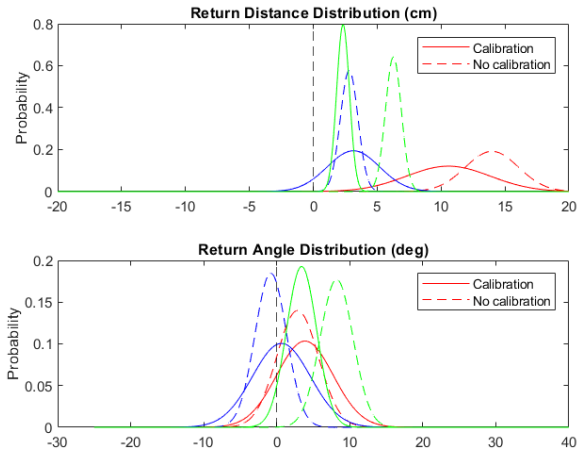


Fig. 11: Calibration against no-calibration in identical paths (60 tests total

with the light sensors. This was motivated by the popular use for robots for autonomous exploration, for which the ability to accurately return home is one measure of good performance.

Over the course of the project, we identified key areas for improving the general system, making possible the implementation of markers. We then studied this improvement, and determined the specific use cases and restrictions for the modified system of nodes. The Romi was able to successfully recognise, register and travel between nodes, and to do so throughout the whole trajectory. However, there was an accuracy impact compared to the basic algorithm. When controlling for this, we find that the calibration routine does indeed improve the average performance but reduces consistency.

The algorithm opens the door for non linear traveling along the map, while enabling the Romi to navigate and return home with high accuracy and precision. The findings relative to our hypothesis indicate that this use case can be applied for longer explorations, both in terms of time and distance. In a one-to-one comparison, there is evidence of better accuracy, and the possibility for travel further without the need for supervision.

However,performance sugests that the algorithm may be better suited for longer and more complex explorations rather than point to point movement. A change in application to a task that requires repetitive movement over predefined points of interest could benefit from the node marker system. Further work consequently could assess the capability of a robot to explore nonlinear-maps and conduct tasks using this updated system.

A shortest-path search feature was implemented but remains to be tested, adding the possibility of finding and moving to the closest node regardless of order. Allowing for shortest-route recalibration could further improve results and considerably lower exploration time, allowing the Romi to move around the map finding shortcuts between the markers.

Is important to note that the non-standard use of the light sensor for identifying shiny materials is not without limitations. During the course of experimentation, several problems arose regarding thresholding values, which seem inconsistent due to illumination conditions and even from sensor to sensor. Figure 12 corresponds to the raw readings from one of the sensors. Note that for one of the channels, there is no distinction between the reading on a white surface and the shiny material. Future implementations would benefit from a sensor with a wider range for luminosity or a dedicated sensor for the markers.

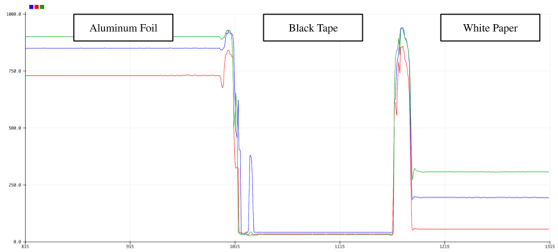


Fig. 12: Incompatible sensor values, seen in some of the available Romi robots

REFERENCES

[1] F. Chenavier and J. L. Crowley, "Position estimation for a mobile robot using vision and odometry," in *Proceedings 1992 IEEE international conference on robotics and automation*, pp. 2588–2589, IEEE Computer Society, 1992.

[2] J. Kim, G. Sharma, and S. S. Iyengar, "Famper: A fully autonomous mobile robot for pipeline exploration," in *2010 IEEE International Conference on Industrial Technology*, pp. 517–523, 2010.

[3] A. Garcia-Cerezo, A. Mandow, J. L. Martinez, J. Gomez-de-Gabriel, J. Morales, A. Cruz, A. Reina, and J. Seron, "Development of alacrane: A mobile robotic assistance for exploration and rescue missions," in *2007 IEEE International Workshop on Safety, Security and Rescue Robotics*, pp. 1–6, 2007.

[4] Kok Seng Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," in *Proceedings of International Conference on Robotics and Automation*, vol. 4, pp. 2783–2788 vol.4, 1997.

[5] F. Lamberti, A. Sanna, G. Paravati, P. Montuschi, V. Gatteschi, and C. Demartini, "Mixed marker-based/marker-less visual odometry system for mobile robots," *International Journal of Advanced Robotic Systems*, vol. 10, no. 5, p. 260, 2013.

[6] A. Paijens, L. Huang, and A. Al-Jumaily, "Implementation and calibration of an odometry system for mobile robots, based on optical computer mouse sensors," *Sensors and Actuators A: Physical*, vol. 301, p. 111731, 2020.

[7] G. Antonelli, S. Chiaverini, and G. Fusco, "A calibration method for odometry of mobile robots based on the least-squares technique: theory and experimental validation," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 994–1004, 2005.

[8] G. Ishigami, E. Pineda, J. Overholt, G. Hudas, and K. Iagnemma, "Performance analysis and odometry improvement of an omnidirectional mobile robot for outdoor terrain," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4091–4096, 2011.

[9] P. O'Dowd, "Ematm0054: Robotic systems." https://github.com/paulodowd/EMATM0054_20_21, 2020.

[10] SHARP Corporation, *SMT, Detecting Distance : 0.5mm Phototransistor Output, Compact Refl ective Photointerrupter*, 10 2005.