Using Bump Sensors to Mitigate Wheel Spin Error in Pololu 3π+ Robot Kinematics

Student Name: Finlo Heath
Student Number 2268276

Student Number: Thomas Rowland
Student Number: 1837499

***Abstract*— This study has explores using bump sensors to estimate when a robot has begun to push an object, providing a novel way to account for wheel spin and the associated odometry errors commonly found in wheeled factory robots. Through two experiments where a Pololu 3π+ robot pushed an object of varying mass along a straight track, a relationship was established between the object mass and the error in odometry. Although there was no significant gain in accuracy by using the Pololu's bump sensors, there was a notable increase in precision across repeated trials, as well as the robot having sensory knowledge that it is pushing an object. In addition, using the bump sensors, the robot was capable of pushing a greater mass from a static in-contact start compared to an algorithm using PID control alone. Full source code, designs and data can be found at: https://github.com/Hinlo/Bump-Sensors-For-Mitigating-Wheel-Slip-Kinematics-Error---Pololu-3Pi**

I. Introduction

A. Context and motivation

There are myriad applications for robots needing to move objects and many different ways of them doing so. For robots pushing objects on the ground, one can envision applications in smart, automated factory settings, surveying nuclear facilities for decommissioning, clearing debris in disaster relief and potentially many more [1]. In the case of automated robots, it is imperative that the additional load does not impede the robot's ability to traverse the environment accurately if the robot is to move the object to the intended position. Accurate onboard odometry would be faster to deploy in factories as it removes the requirement for a centralised location system and the associated infrastructure changes.

This brings us to the challenge motivating this study, wheeled robots can experience wheel-slip when moving on a given surface. Wheel-slip is when the wheels spin without causing any physical motion of the robot and occurs when the friction between the wheel and the floor is too low for a given driving force [2]. This issue is significantly exacerbated when pushing an object, as friction between the object and the floor provides an opposing force to the robots driving force.

This study uses a Pololu 3π+ two-wheeled robot [3]. The kinematics of this robot, are based on the encoder counts registered when the wheel turns [3], meaning the robot will register a change in its perceived position. The robots perceived position is inherently linked to the rotation of the wheels, this will cause errors should the wheels slip as the wheels will rotate without any change in the robots actual position. This error in the robot's kinematics can cause it to significantly undershoot when directed to push an object a given distance. This would severely limit it's real world utility. For example, a warehouse robot is little use if it never accurately moves objects to a directed location. The requirement therefore, is for the robot to have some method of mitigating any error which occurs due to wheel-slip, allowing it to maintain accurate kinematics and thus accurately move as required in the real world.

B. Literature Review

Several other studies have looked to find solutions to the problem of wheel slip in warehouse robots. The problem is not localised to the traditional wheel. Galati et al. [4] explores how certain dynamic effects such as slippage cause errors to the pose and location accuracy with an omnidirectional robot using mecanum wheels. They chose to categorise these errors and perform corrections dependant to the type of material the robot was traversing.

Smieszek et al. [5] investigated how increased load can effect the wheel rolling radius and slip in small mobile platform robots. It is stated that for odometry to be accurate, rolling radius, wheel base and non-slippage needed to be constant. This study looked to determine a relationship between increased load and increased slip so that a regular correction could be applied dependant on the load.

Our research looks at a new way to determine when the object has started moving and correct for wheel slip at the start of the operation.

C. Background information

1) Friction: A force is required to move one object against another. Static friction (F_s) is the force required to initiate a sliding motion and kinetic friction (F_k) is the force required to maintain that sliding motion at a constant velocity [6]. The difference between static and kinetic friction is fundamental to the friction phenomena with the static friction being higher than kinetic friction due to the asperity - microscopic unevenness. F_k and F_s can often differ by several tens of percent [7]. Once an object is in motion it slides over the asperities and hence the kinetic friction is lower.

2) Pololu 3π+: This is a small but versatile user-programmable robot. It is controlled by an ATMEGA32U4 microcontroller that is programmable through the Arduino environment and includes several inbuilt peripherals [3]. These peripherals include 2 bump sensors located on the front of the robot which are the focus of our investigation.

The bump sensors are active sensors that consist of an 2 infrared (IR) emitter LEDs paired with a phototransistor. The sensors detect IR light reflected off of the inside of the bumper skirts as well as ambient light [3]. The 2 bumper skirts allow a depression of $\approx$ 1.5mm and hold their elastic potential energy when compressed. Contact with the bumpers is inferred by the time it takes for a capacitor in the sensor system to discharge. The longer capacitor discharge time, the less IR light is reflected to the sensors which represents greater bumper skirt compression.

3) Key Assumptions: Three key assumptions are made and form the premise of our solution to mitigate wheel-spin error as the robot pushes an object:

- We assume that contact force between robot and object increases until the point of motion and then remains below this maximum value after motion begins.
- We assume that, during motion, the PID controller does not vary the Power supplied to the motors by a large enough value to invalidate the above assumption.
- We assume the bump sensors can be modelled as an ideal spring, thus obeying Hooke's law [8].

D. Hypothesis Statement

Considering the many applications of pushing robots, and the issue of wheel-slip reducing the accuracy of the $3\pi+$ odometry, we postulate the following hypothesis:

> *When starting to push an object, the bump sensors will reduce kinematic error by providing a sensory trigger to mitigate the effects of wheel-spin. This effect will be more significant when pushing a greater mass, as friction is proportional to object mass, so there is a larger opportunity for wheel-spin before it is overcome. Error mitigation will extend to impacts with stationary objects as well as starting the robot in stationary contact with them, reducing the error in real world distances travelled by the robot when tasked with pushing an object a defined distance.*

This paper first describes the implementation three algorithms for two experiments in section II, including our investigation into the relationship between contact force and bump sensor measurements in section II-A. Our experimental procedure is outlined in section III with dimensional diagrams of the experimental set up for reproducibility. Our results are displayed and qualified in section IV. Finally, the implication of our results and further work are discussed in section V.

II. Implementation

To investigate whether the bump sensors can be used to mitigate wheel-slip error, three different algorithms were written and used as shown in Fig. 1. The base software (Software 1) was created first. This uses a straight line PID controller [9] to ensure linear motion at a chosen demand speed, the robot travels forward until it's x position kinematics reading reaches 30.0cm (co-incident with robot forward motion). The PID controller proportionality constants are identical for each algorithm developed.

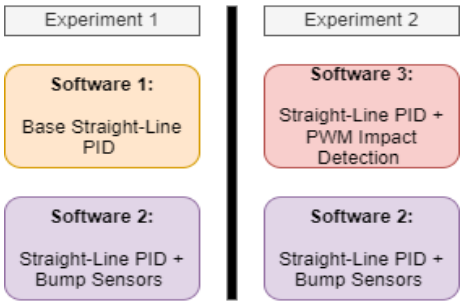

Fig. 1. A high level overview of the different software algorithm used in each experiment. The same bump sensor algorithm is used as the second condition in each experiment. Impact detection using PWM values is introduced in experiment two for the first condition as a requirement of the experimental design.

When the robot is initialised in stationary contact with an object, the PID controller increases power to the wheel motors until the encoder count rate matches a given demand speed. This increase in power amounts to a greater pushing force on the object, until the robot moves the object at this demand speed. If the wheels slip during this process, the base code has no sensory information to distinguish this from real distance travelled by the robot.

For our second condition (Software 2), the base software was built upon to include bump sensor readings. A metric analogous to contact force was calculated and used to infer the moment at which the robot started to move the object. The derivation of this metric is discussed in detail in section II-A. We assert that, until the compression limit of the bump sensors is reached, there is a clear experimental proportionality between the contact force applied to the to the bump sensors and the capacitor's discharge time. Thus we will refer to the contact force hence forth, with the understanding that increases or decreases in contact force are inferred from increases or decreases in measured discharge time.

As explained in section I-C1, the force required to overcome the static friction of an object is greater than that required to overcome the kinetic friction. For the bump sensor algorithm, when a new maximum contact force is measured, the recorded position of the robot in the kinematics is reset to zero. Any wheel spin before the object starts to move will be ignored, providing a more accurate real world pushing distance.

In our second experiment, we investigate the effect of impact with the object on kinematic accuracy, as detailed in section III-C. This necessitates that the algorithm without active bump sensors be provided a method of impact detection, leading to the creation of our final algorithm (Software 3). As mentioned a priori, when incident on an object, the PID controller increases power to the motors in order to maintain the demand speed. As such, we simply set a threshold power after some preliminary tests which, if reached, indicates that the robot has had to supply significantly more power to the motors to maintain speed

and thus has impacted the object. The kinematics are reset at this point and the robot attempts to travel the desired $30.0cm$. Our three algorithms are summarised in Fig. 2

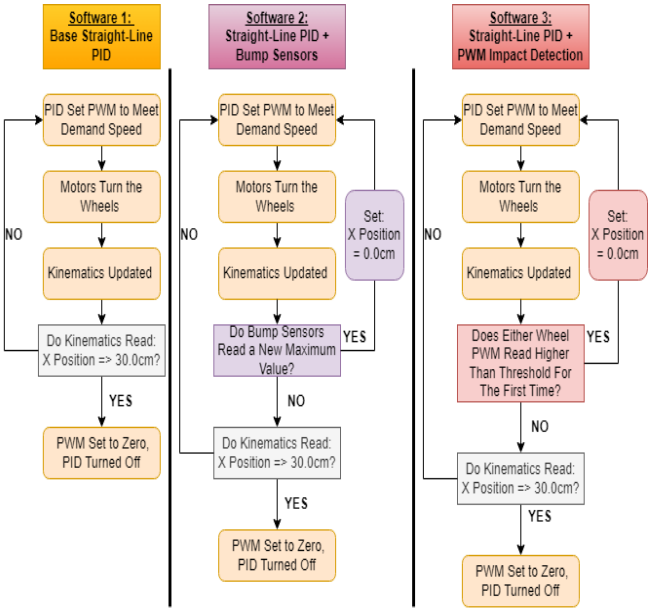

Fig. 2. The logic flow behind the three algorithms, used in our two experiments as shown in Fig. 1.

The full code for each algorithm can be viewed in our GitHub repository [10], the URL is provided in the Abstract.

A. Investigation of Robot-Object Contact Force

This section outlines the steps taken to discern whether the robot-object contact force could be derived from the capacitor discharge time measurements provided by the bump sensors. Fundamentally, as we investigated only a linear relationship between capacitor discharge time and contact force, our algorithm makes no numerical distinction from simply using the maximum discharge time to discern when the robot has started to move the object. The overall aim of the investigation is to use the bump sensor measurements to mitigate wheel-spin error. Thus the integrity of the investigation stands, despite being unable to verify the accuracy of the quantification of contact force between the robot and object. A study to fully articulate the relationship between bump sensor capacitor discharge time and contact force of an object comprises part of the further research detailed in section V. The paragraphs herein detail the investigation and final metric used in the bump sensor algorithm.

The bump sensors are elastic, and as stated in section I-C3, we have modelled them as a spring with the compressive force described by Hooke's law [8],

$$F = kx$$

where k is the spring constant and x is the linear displacement of the bump sensors. By Newton's third law [11], an equal an opposite force of $-kx$ is applied to the object compressing the bump sensors.
The value of k can be found experimentally by orienting the robot vertically and placing a known mass on the bump sensors that is just large enough to compress them fully. Under these conditions, the compressive force F is given by the object's acceleration due to gravity [12]:

$$F = mg, \quad g = 9.81ms^{-2}$$

Equating this to Hooke's law and rearranging for the spring constant, one finds:

$$k = \frac{mg}{x}$$

There is a clear proportionality between the compression of the bump sensors and the measured time for their capacitor to discharge. Given this displacement is directly proportional to the force applied, it follows that there is a proportionality between the force applied to the bump sensors and time to discharge. However, during the scope of this study we were unable to discern the exact nature of this proportionality, whether linear, quadratic, cubic or otherwise. This is because the bump sensor measurements oscillate naturally under constant lighting, vary greatly with changes in ambient lighting and are no longer proportional to the contact force once the compression limit of the bumpers has been reached. Investigations into calculating the contact force using the constant acceleration equations of motion [12]were made, but it is difficult to ensure constant acceleration or measure the interval between starting and reaching the demand speed. When modelling the discharge time - contact force relationship as linear, $F = ct$ where c is a constant and t is the discharge time, averaged across the two bump sensors and calibrated to offset background measurements. We find that the previously derived equation of spring constant can now be adapted to the "discharge" constant, c:

$$c = \frac{mg}{t}$$

using a mass which just fully compressed the bump sensors, $m = 0.164kg$, $t = 9.64 \times 10^{-4}s = t_{measured} - t_{background}$, $g = 9.81ms^2$, we calculated $c = 1700kgms^{-3}$. t is the discharge time calibrated to remove background measurements, $t_{measured}$ is the actual measurement read by the bump sensors, $t_{background}$ is the background measurement read when the bump sensors are not in contact with anything. Testing this our bump sensor algorithm, one finds contact forces of between 0.1 - 2N. This is not unreasonable, but as stated above we were unable to validate this experimentally. In our bump sensor algorithm, we use the Contact Force:

$$F = ct = 1700 \times (t_{measured} - t_{background})$$

to discern whether the robot as begun to push the object, with the understanding that the greatest value of F will be measured at the instant before the static friction is overcome and the object starts to move.

III. Experiment Methodology

To test our hypothesis we devised an experimental routine whereby the Pololu $3\pi+$ robot pushes an object of known mass in a straight line. For each experiment the two implementations are compared by how accurately they

could push this mass to a specified distance. The mass is then be incremented to the upper limit of the robots pushing capabilities to see how the use of the bump sensors affected the robots perceived distance accuracy.

It was decided that the pushed object should be the Pololu $3\pi+$ robots own box and rice would be added inside to increase the mass evenly without changing object surface area or volume. Early testing made it clear that it is near impossible for the robot to push this box in a straight line. A simple track was designed with the following objectives:

- To provide a smooth and even surface for the robot to push the object.
- To ensure that the box is pushed in a straight line.
- Any friction from any side rails must be consistent across all trials and at acceptable level.
- The design must facilitate easy measurement of distance and a quick reset of the experiment.

A. Track Design

To design a track to the above criteria corelDRAW graphics suite [13] was used with a laser cutter for manufacture. This ensured a higher level of accuracy over hand machining methods and ensured any cuts and markings would be straight. The material used was $12mm$ thick poplar plywood, the separate parts were then laser cut and slotted together. The parts consisted of a main track bed and two sides, and end piece. The sides were inserted into the bed with enough separation to allow the box to slide unhindered along the track but tight enough to ensure that course was straight and that no turning moment could occur. To further reduce the friction from the sides, the end piece was inserted between them at the end of the track to give the sides a very small angular deflection ensuring that only the lower corners of the box where in contact with the sides. A track length of $45cm$ with a centre line and centimetre markings is etched into the surface for ease of measurement and alignment. The track is displayed in Fig. 3.

B. Experiment 1 - In-contact Pushing

The first experiment was designed to determine the effectiveness of the bump sensors to account for loss of traction in the initial stages of pushing the box. As explained above in section I-C1 it was assumed maximum measured force would be just before motion occurs. The PID only algorithm and bump sensor algorithm are both tasked with travelling $30cm$ along the track. This distance would give enough opportunity for the any odometry error to be noticeable. Each trial will consist of the Pololu $3\pi+$ robot pushing the stationary box, from rest, a distance along the track. When the robot kinematics measure that it has travelled $30cm$, it will stop. The real world distanced travelled will then be measured to determine the accuracy of the trial.

Upon completion of the track construction, preliminary readings where taken to establish the maximum mass that the robot would be able to push. This was found to be approximately $280g$, of which $53g$ was the box and the remaining $227g$ was rice. The experiment was run with different mass categories ranging from $55g$ to $280g$ in $25g$ increments with a preliminary "control" run of $0g$ (without the box). As shown in Fig.3 each trial started with the Pololu's rear roller ball being placed on the cross at 0cm and measurements where taken from where the roller ball was in contact with the track after the trial. At the start of each trial the box was placed just at the point of contact with the bumpers. 10 results were collected for each mass category. This was repeated for both implementations so that a comparison of methods could be conducted.

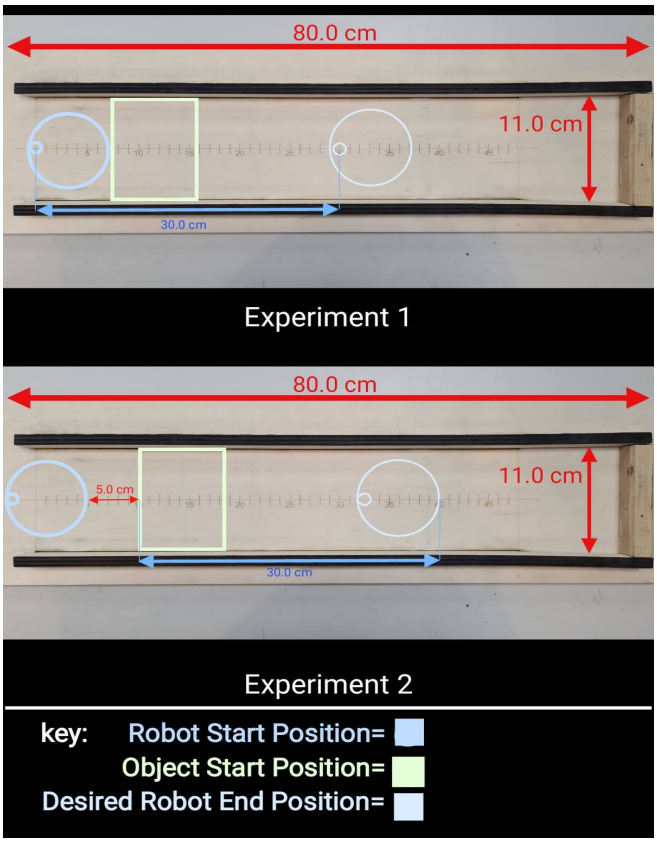

Fig. 3. A top-down view of the track illustrating the start and end positions for each of the two experiments. The smaller concentric circle marks the rear roller ball of the robot to indicate it's yaw orientation.

C. Experiment 2 - Impact Pushing

The second experiment was designed to determine which implementation would be performs better should the pushed object start not in contact with the robot. The front edge of the robot was placed at the $5cm$ mark and the back of the box was placed at the $10cm$ mark as shown in Fig.3. The robot will move forward $5cm$ and impact the box. From this point, the robot will aim to push the box a further $30cm$ to the $40cm$ mark on the track where it will stop. Distance measurements were taken from the front edge of the bumpers by removing the box once the robot has come to rest. The mass range begin with a "light" $55g$ trial, then continued from $155g$ to $280g$ in $25g$ increments. 10 trials were taken at each mass with each algorithm.

D. Discussion of Variables

- **Controlled Variables**: Once the track had been cut any contact points between the box and the track were sanded down to a smooth and even finish. Friction from the side rails was minimal due to the outward

angular deflection. The bottom edge of the box was kept in constant and even contact with the side rails to prevent any yawing movement from the box. The orientation of the box was kept the same to ensure testing was uniform. Battery level was kept constant between the experiments with tests completed with both implementations for the same mass category before increasing the mass. Testing for both experiments took place on the same desk, checked with a spirit level to ensure it was level. Constant bright ambient lighting was used throughout.

- **Independent Variable**: This is the object mass, comprised of the Pololu $3\pi+$ box and varying amounts of rice. The mass was measured on digital scales to an accuracy of $\pm 0.5\,\mathrm{g}$. As the mass of the object increases the friction between the track and the box increases by the relationship $F = \mu N$ [6] where F is the frictional force, μ is the friction coefficient and N is the normal to the track surface.
- **Dependent Variable(s)**: In both experiments the distance that the robot travelled was measured as shown in Fig.3. This was then compared to the control to determine how much the box mass impacted the odometry. Early trials indicated that any loss of traction at the point of motion resulted in odometry error of a few centimetres. Therefore a resolution of $0.25cm \pm 0.25cm$ was deemed sufficient. This was measured by eye with reference to the 1cm markings on the track surface.

IV. Results

Figures 4 & 5 display the results from our first and second experiment respectively. So as to more easily compare loss of accuracy in push distance as the object mass increases, between the two algorithms, Fig.4 is calibrated vertically to align the results of the control trials between the two algorithms. The reader should take note therefore, that the vertical axes differ between the two algorithms.

Both Fig. 4 and Fig. 5 show a clear downward trend in distance measured as the mass of the box is increased. This shows that pushing a larger mass causes increased kinematics error using both algorithms.

In each algorithm, the robot's kinematic calculations and variables are the same. Investigations into why the PID Algorithm falls short of the required distance in the control trials were not made, as the study is concerned with comparative accuracy between the control and object pushing trials - as opposed to the absolute accuracy between the two algorithms. Absolute accuracy can be improved by separately calibrating the set wheel circumference for each algorithm, but this could not be performed in our study as it would invalidate comparison between algorithms.

In Fig. 4, the bump algorithm, displayed in red, was more consistent in distance travelled throughout the experiment. It also performs significantly better as the robots maximum moveable object mass is approached; this is evidenced by the trials taken with object mass of $255g$. At this mass,

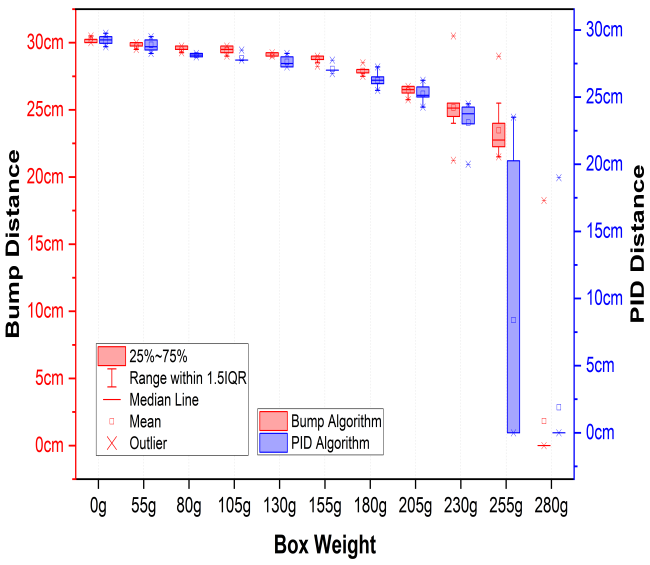

Fig. 4. Experiment 1 - The above graph illustrates how increasing box mass affected the robot odometry when starting in stationary contact. The left and right Y axis' are set for the "bump" and "PID only" algorithms respectively. Both Y axis' are to the same scale, but shifted as to align the control runs to better illustrate how each algorithms compared with each other.

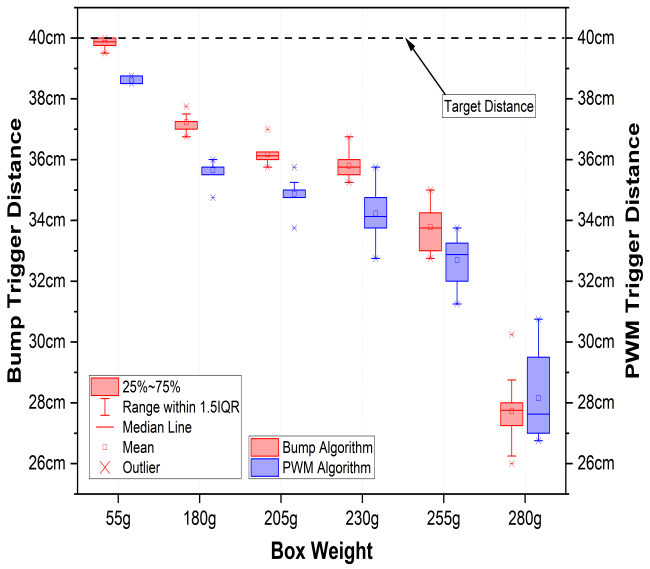

Fig. 5. Experiment 2 - This graph shows how box mass affected odometry without starting in contact with the robot. The Y axis is not shifted to align control trials in this graph, as a true massless control trial was not possible.

the PID Algorithm only managed to move the box off of the start line 40% of the time compared to 100% with the bump sensor.

In Fig. 5, while the bump sensor algorithm again provides more consistent results than its PWM counterpart, there is less significance in overall accuracy of either algorithm. This implies that the bump algorithm as a method of reducing kinematics error is less robust when the moving robot impacts a stationary object. This could be due to the momentum of the robot reducing the time to move the object and providing less opportunity for wheel slip before the object begins moving. Both algorithms in the impact experiment were still able to move the box 100%

of the time at $280g$.
Across both experiments, wheel-slip occurred in some trials whilst the robot was moving the object, a phenomena that none of the algorithms used can detect. This likely had a significant impact on distance travelled in many of the trials. Thus, providing methods to detect wheel-slip in transit comprises one of our key areas of future study.

V. Discussion and Conclusion

Our hypothesis in section I-D, states that the bump sensors will mitigate wheel-spin error occurring before static friction has been overcome, leading to improved kinematic accuracy and thus more accurate real world travel distance. Our results clearly show that this hypothesis is invalid. In both experiments and across each object mass, the bump sensor algorithm did not travel significantly closer to the target distance compared to the PID only algorithm, when accounting for the disparity in control distance for each algorithm.

We postulate this is because there is no significant wheel-slip occurring before the object begins to move. Visually, during experiments, the robot would be able to start pushing the object without slipping, or the wheels would slip and the object is never moved - this is the case for the $280g$ trial in Fig. 4. There were no cases where the wheels re-gained friction after starting to slip, before object motion occurred. The main wheel-slip error seems to be slips imperceptible to the human eye during motion of both object and robot, with more slip occurring for larger object mass.

The results do validate the assumptions made in section I-C3, as the robot never overshot the target distance, meaning the contact force during motion did not exceed the maximum value measured as the object begins to move. This also implies that our model of the bump sensors as an ideal spring is valid, as were this not the case, the lower contact force during motion could still have produced greater bumper compression and thus a higher capacitor discharge time measurement.

The results show that use of the bump sensor algorithm does not produce significantly worse performance. It also increases the proprioceptive abilities of the robot, as it now has sensory information that it is pushing and object, as well as when it has started to move the object. This increased robot intelligence has value in the context of our motivation, providing better warehouse robots; the PID only algorithm cannot do this. This could be adapted to trigger different states in a finite state machine, allowing the robot to change behaviour once it starts and stops pushing an object. The same could be said of the impact detection algorithm, though it cannot estimate contact force and thus mass of the object being pushed. Both are therefore useful expansions to PID only control, but our bump sensor algorithm significantly increases the contextual intelligence of the robot for warehouse use, without decreasing the performance, improving it's utility for this application.

It is clear that two elements of this study would most benefit from further investigation. A quantitative analysis of the relationship between bump sensor capacitor discharge time and the compressive force applied would provide more accurate force measurements and therefore more accurate inference of when the robot has started to move an object. It would also allow calculation of the object mass, useful information in a warehouse environment, for example, verifying the robot is pushing the correct object.

Secondly, if localised maximum contact force measurements are recorded in addition to the overall maximum value, the system could speculate wheel-slip occurrence whilst moving the object. This would require detection of a local maximum force value and reset of the robot's internal position value to that taken at the time of the previous corresponding local minimum force, rather than to the starting position. Given that the robot experienced wheel-slip during pushing, these changes could lead to a greater significance in performance between the algorithms tested in this study, increasing the viability of bump sensors measurements for mitigating wheel-slip error.

References

[1] Stüber J, Zito C, Stolkin R. Let's Push Things Forward: A Survey on Robot Pushing. Frontiers in Robotics and AI. 2020;7. Available from: https://www.frontiersin.org/articles/10.3389/frobt.2020.00008.

[2] Pololu 3pi Robot User's Guide. Pololu;. Accessed Dec. 08, 2022 [Online]. Available from: https://www.pololu.com/docs/0J21/all.

[3] Pololu - 3PI+ 32U4 OLED robot - Standard edition (30:1 MP motors), assembled. Pololu;. Accessed Dec. 03, 2022 [Online]. Available from: https://www.pololu.com/product/4975.

[4] Galati R, Mantriota G, Reina G. On the origin of why static or breakloose friction is larger than kinetic friction, and how to reduce it: the role of aging, elasticity and sequential interfacial slip. Sci Rep. 2022 Nov;12(19608).

[5] Smieszek M, Dobrzanska M, Dobrzanski P. The impact of load on the wheel rolling radius and slip in a small mobile platform. Autonomous Robots. 2019 12;43.

[6] Lorenz B PBN. On the origin of why static or breakloose friction is larger than kinetic friction, and how to reduce it: the role of aging, elasticity and sequential interfacial slip. J Phys Condens Matter. 2012 Jun;24(225008).

[7] K Hashiguchi SO. Constitutive equation for friction with transition from static to kinetic friction and recovery of static friction. Int Journ of Plasticity. 2008 mar;24(11):2102-24.

[8] Young H, Freedman R, Ford ACA. Figure 6.18 - Force Required to Stretch a Spring. In: University Physics with Modern Physics. 14th Edition. Pearson Education, Edinburgh Gate, Harlow, Essex, England; 2016. p. 207-8.

[9] Knospe C. PID control. IEEE Control Systems Magazine. 2006;26(1):30-1.

[10] Heath F, Rowland T. Hinlo/Bump-Sensors-For-Mitigating-Wheel-Slip-Kinematics-Error—Pololu-3Pi. GitHub;. Accessed Dec. 03, 2022 [Online]. Available from: https://github.com/Hinlo/Bump-Sensors-For-Mitigating-Wheel-Slip-Kinematics-Error---Pololu-3Pi.

[11] Young H, Freedman R, Ford ACA. 4.5 Newton's Third Law. In: University Physics with Modern Physics. 14th Edition. Pearson Education, Edinburgh Gate, Harlow, Essex, England; 2016. p. 140-5.

[12] Young H, Freedman R, Ford ACA. 3.3 Projectile Motion. In: University Physics with Modern Physics. 14th Edition. Pearson Education, Edinburgh Gate, Harlow, Essex, England; 2016. p. 99-101.

[13] . CorelDRAW Graphics Suite;. Accessed Dec. 03, 2022 [Online]. Available from: "https://www.coreldraw.com/en/product/coreldraw/?topNav=en".