Intentional Abuse Avoidance System (IAAS)

By - Jakob Sörstadius, Lukas Larsson, Mahin Garg, Robin Ackermann

Abstract

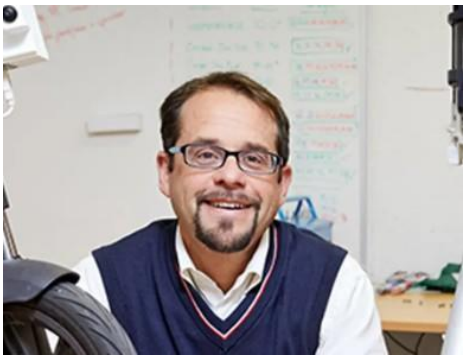
Shared micro-mobility vehicles, particularly e-scooters, have become increasingly popular in urban areas, leading to a notable rise in usage and associated accidents. This study explores the development of an anti-abuse system designed to detect and mitigate skidding and burnout behaviors in e-scooters to enhance rider safety and reduce wear and damage. Utilizing Hall effect sensors on both wheels, the system detects differences between normal riding, standstill, skidding, and burnout scenarios, notifying riders through visual and auditory signals when skidding.

The project involved prototyping with 3D-printed mounts for sensors, developing software for controls and conducting tests to assess system reliability. Results showed effective detection capabilities, although the system experienced a number of “false positives”, which could impact user trust. Enhancements in sensor accuracy and data processing are proposed to address these issues. The study demonstrates the feasibility of implementing a cost-effective anti skid system for e-scooters, contributing to improved rider compliance and overall safety in the growing micro-mobility sector.

Acknowledgements

We would especially like to thank Rahul Rajendra Pai for going beyond what regular teaching assistants do. We really appreciate the effort you put in for us, especially with access to Voi e-scooter. We would also like to thank you for your valuable insights and many laughs during the formulation of this project. Good luck with your future studies.

Another acknowledgement also goes to Marco Dozza for putting this excellent course together. It's very rare to see such passion for making the student experience this good. We see and appreciate the effort you made to arrange all the activities. From flying in teachers from the Netherlands to letting us ride SEK 100,000 bikes to teaching important concepts both theoretically and practically, we really like to say that we appreciate it, and hope that this course will continue to inspire students in the future.



Content

Introduction

Since a few years back, the introduction of shared micro mobility vehicles has reshaped the way people move in cities. These vehicles primarily consist of E-scooters that can be rented and parked anywhere, and just requires the user to have a smartphone and credit card. Between January and August 2019, 4 million E-scooter trips were made in Sweden [1]. A study by Folksam shows that even though E-scooters are a small part of total trafficants in Sweden, they are overrepresented in accidents. The study also found that most reported accidents involved a "single" accident type, meaning no other vehicle or pedestrian was involved [2]. Riding an e-scooter requires coordinating multiple tasks. If the rider locks any of the wheels (also called skidding), it can cause a sudden loss of grip, which is particularly dangerous.

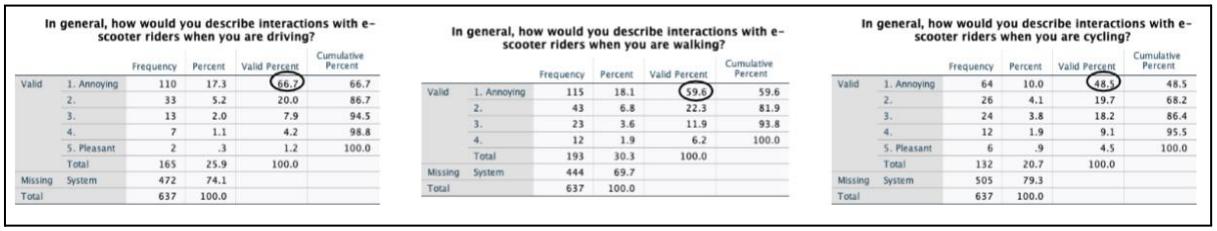
E-scooters are not only used for transportation. Studies have shown that a significant number of e-scooter trips are made for leisure, which tend to be more dangerous [3]. The US E-scooter company Bird reports that some riders skid on purpose. They have therefore implemented an anti skid system which they claim to “Elevate E-scooter safety and compliance” [4]. If the user is either experiencing or performing an uncharacteristic amount of skidding Bird is alerted, enabling quick response from the company. If the skidding is determined to be intentional Bird will first trigger a series of notifications to the rider. If the behavior continues, rider access can be removed.

Consequences of Skidding and Burnouts:

Figure 1 Worn out e-scooter tires.

E-scooters are single track vehicles and rely on concave tyres in order to lean and steer. The tyres wear during normal riding, but rapid braking is particular wearing on the tyres. This causes a flattening in the center of the tyre, which reduces its concaveness and therefore its handling capabilities. Tyre wear also reduces its thread pattern, which makes the tyre less adhesive on wet and uneven surfaces.

Apart from this, there are studies showing that e-scooter riders are not accepted (see Figure 2) that well in many places, with a majority of road users (motorcyclists, cyclists, pedestrians) finding e-scooter riders as “annoying”. Some reasons for this may include unsafe driving maneuvers like skids and burnouts, which may also contribute towards more accidents and lead to negative perception of e-scooter users.



In general, how would you describe interactions with e-scooter riders when you are driving?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1. Annoying	110	17.3	66.7	66.7
	2.	33	5.2	20.0	86.7
	3.	13	2.0	7.9	94.5
	4.	7	1.1	4.2	98.8
	5. Pleasant	2	.3	1.2	100.0
	Total	165	25.9	100.0	
Missing	System	472	74.1		
Total		637	100.0		

In general, how would you describe interactions with e-scooter riders when you are walking?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1. Annoying	115	18.1	59.6	59.6
	2.	43	6.8	22.3	81.9
	3.	23	3.6	11.9	93.8
	4.	12	1.9	6.2	100.0
	Total	193	30.3	100.0	
Missing	System	444	69.7		
Total		637	100.0		

In general, how would you describe interactions with e-scooter riders when you are cycling?

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1. Annoying	64	10.0	48.5	48.5
	2.	26	4.1	19.7	68.2
	3.	24	3.8	18.2	86.4
	4.	12	1.9	9.1	95.5
	5. Pleasant	6	.9	4.5	100.0
	Total	132	20.7	100.0	
Missing	System	505	79.3		
Total		637	100.0		

Figure 2. Survey results for acceptance of e-scooters. [5]

Due to the absence of technical insight on how the Bird anti-skid system works, and also the lack of insight on how much this system cost, we decided to implement a low cost and simplistic version of this anti-skid technology, and also implement burnout detection here.

The main objective of the IAAS system is to detect the following scenarios and identify them appropriately

- if the front wheel is locked while the back wheel is running (burnout),
- the back wheel is locked while the front wheel is running (skidding),
- both wheels are turning (normal riding),
- both wheels are still (standstill)

Then after detection, it must warn the user in case skidding or burnout is detected.

Background Technology

A. Hall effect sensor

Hall effect sensors work by becoming active in an external magnetic field, and then it is able to sense if the position of the magnet is close or far via varying magnetic field strength. The output signal that comes out from a Hall effect sensor represents the density of a magnetic field around the device. Hall effect sensors all have a thin piece of semiconductor material inside them, which passes a continuous electrical current through itself to generate a magnetic field. [6]

B. Arduino

The Arduino Uno R3 is a microcontroller. [7] The Arduino board can be connected to the computer via USB and programmed. It can be connected to electronic components like sensors and actuators. It is small and can be attached to the scooter.

Methodology

We started the project by first defining the scope of the project. For the initial prototype, we aimed to detect skidding and alarm the user via a light sensor and a buzzer. We chose to work on a Voi E-Scooter due to ease of access and some technical insights through the course.

We started by identifying the components that will be needed for the same. The list of components is as follows -

1. Hall Effect sensor (2)
2. Breadboard (1)
3. Arduino uno R3 (1)
4. Connection cables (15)
5. Neodymium magnets (2)
6. Resistor (2)

Mechanical

We also needed 3D printed parts to mount the magnet on the front and rear wheels. We start this process by first having rough measurements for the wheel diameters for the front and rear wheels of the e-scooter. We make an initial 3D print for both the front and rear wheels, and then we notice the slight fitting problems for the rear wheel in terms of part thickness. We also rectify our initial wheel diameter values with inputs from our course TA and his access to CAD models of the said e-scooters. In this way we were able to perfect the fitting of these mountings. The final CAD drawings for the front and back wheel mounting are mentioned below.

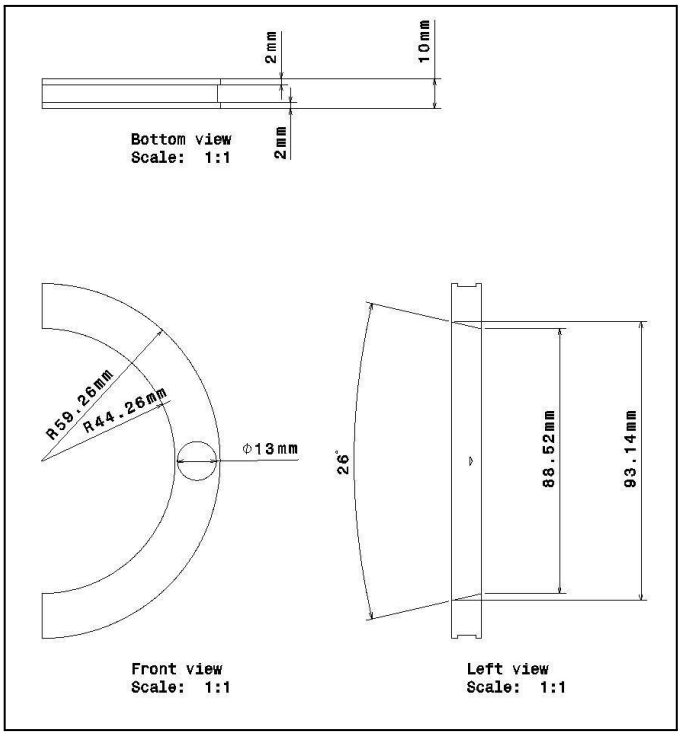


Figure 3. CAD drawing for front wheel mount.

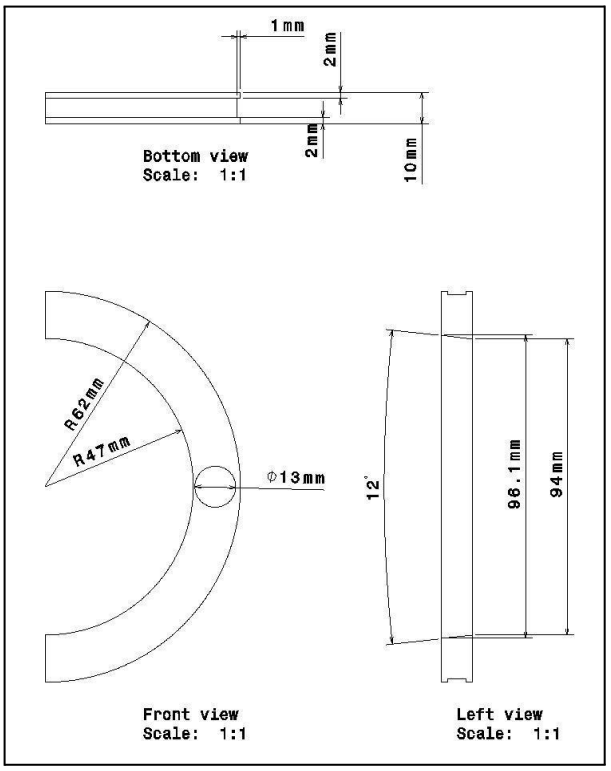


Figure 4. CAD drawing for rear wheel mount.

Both circular mountings were made in 2 halves, and then joined together using hot glue for ease of access in case of further modifications. For the 3d prints, we used PETG material for increased durability, with a 2 mm layer thickness and 15% infill for both wheels. We also added 2 magnet holes in both these mountings as an added redundancy in system implementation. Next we needed a mounting plate to place the arduino and breadboard on. Since this part did not need precise fitting, so instead of 3d printing another part, we searched the recycled PETG container for any parts (With the main hope of being sustainable and saving printing time and material). Luckily we found a part that fulfilled our requirements exactly so we used the same.

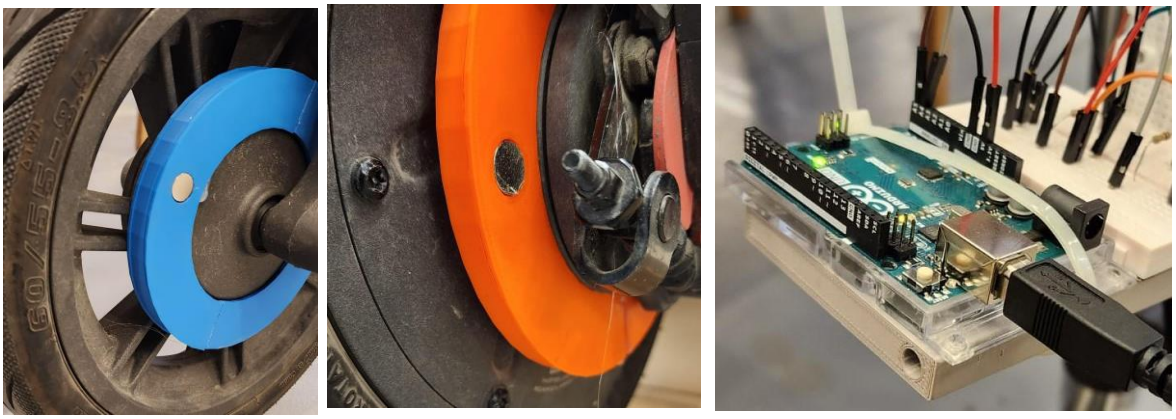


Figure 5. Mounted parts.

Software Development and Electronic Integration

The software development began with creating an overview of the system's basic functionality, see figure 6. The hall sensors were set up and the information received from these was examined. It was decided that initially the system should rely on determining if a wheel is locked or turning. Reliably calculating the speed from the sensors alone was deemed both unreliable and unnecessary for the purpose of the project.

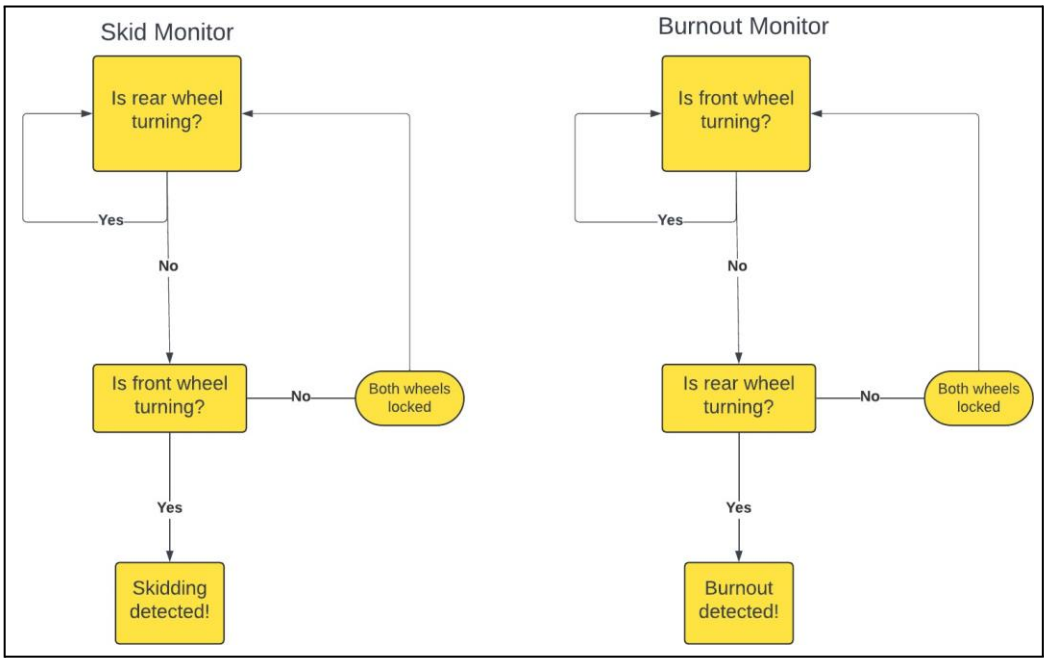


Figure 6. Basic structure of program.

Interpreting Sensor Data

The first step in the software development was to interpret the signals from the hall effect sensors to register when one full rotation of the wheel has occurred. The signals from the hall effect sensors are read as an analog value at a constant interval. If a magnetic field is detected by the sensor the value registered at the controller will be 0. This means that the values read from the sensor will be 0 for the whole duration while the magnet is present (see figure 7). In order to avoid misinterpreting multiple readings during one passage as additional rotations a couple thresholds were put into place.

If a 0 is detected a value of 10 or larger needs to be detected before another rotation can be registered. This ensures the magnet exits the hall sensor range before a rotation is recorded.

After the magnet is registered the following five readings are unable to trigger another rotation.

Calculation behind waiting five readings:

- Diameter of the wheel is 0,29m
- Circumference is 0,29 * □ ≈ 0,91m
- Each reading is done with an interval of 20ms
- Assumed top speed of scooter is 30km/h = 8,33 m/s
- Time for one rotation at max speed: 0,91 / 8,33 ≈ 110ms

- Lowest number of readings during one rotation = 110/20 = 5,5 -> 6 readings

The calculation shows that the sixth reading is the first reading to possibly register another interval assuming max speed is 30 km/h. Therefore the following five readings after a detection were decided to not be able to trigger another rotation.

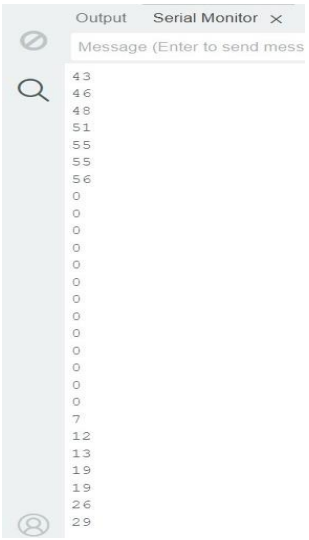


Figure 7. Sensor data.

Determining State of Wheels

The main challenge with determining the state of the wheels was to decide when the wheel should be determined locked. When the wheel is in rotation it continuously registers the magnet, but if the sensors aren't triggered, then how long should the system wait until assuming the wheel is locked?

This scenario has a few conflicting interests. On one hand the longer the system waits the more certain it is that the wheel is locked/stationary instead of rotating slowly. On the other hand, time is an important factor. The faster the locking of the wheel is detected the faster a potential skid or burnout would be detected. Faster locking also would enable shorter skids to be detected, especially important when trying to identify an unintentional skid.

Calculations to determine time since last trigger until the wheel is assumed locked:

- Assumed lowest riding speed 6.5 km/h = 1.81m/s
- Wheel circumference = 0.91m
- Time for one rotation at 6.5 km/h = 0.91 / 1.81 = 0.50s

Based on the calculation it was decided to set the threshold for assuming the wheel is locked to 0.5 seconds. This would hopefully ensure most skids are detected while still trying to maintain reliability at lower speeds.

Skid and Burnout Detection

The next step was to monitor both wheels and decide when to trigger a skid or burnout. The system's definition of a skid would be that the rear wheel is locked while the front wheel is still in rotation. This created another issue stemming from the two wheels not being aligned.

Despite both wheels locking simultaneously, it still might take some time for the front to reach the same state.

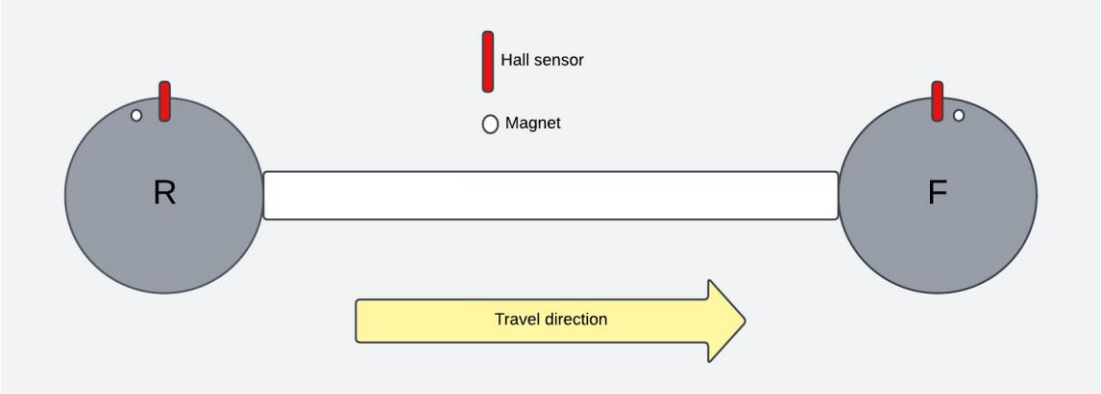


Figure 8. Schematic overview of detection system.

The “worst case” scenario is shown in the figure above. Let’s assume the scooter has been traveling close to the 6.5km/h limit up until this point and now both wheels lock completely. The rear wheel will almost immediately enter the locked state, meaning that 0.5s has passed since the last trigger. The front wheel would however need close to the whole duration of 0.5s before entering the same state, creating the impression of a skid despite both wheels locking/stopping simultaneously. This prevents the system from immediately deciding if there is a skid or not when detecting the rear is locked.

To account for this scenario the monitor for both skid and burnout was designed to observe the opposite wheel for 0.5s when a lock is detected. If the opposite wheel hasn’t entered a locked state after this time, indicating both wheels have stopped correctly, the system alarms that a skid/burnout has occurred. The alarm was implemented as a flashing LED and a buzzer continuously beeping for the duration of the skid/burnout, but for a minimum of 0.5 seconds.

Fine Tuning

After the implementation of the previously described functionality the system was verified and fine tuned. While there was theory behind decisions during development, the potential for refinement in order to improve the system in a more practical setting was quickly apparent. The main problem to address was to deal with false positives which occurred quite frequently, especially from the rear. The false positives were the hall sensors registering a magnetic field while the magnet wasn’t present.

First off some countermeasures were already in place, for example the minimum number of readings between registrations. Additionally, resistors were placed between the output from the sensors and the arduino to increase the necessary strength of the signal to have an effect. Additionally the cables between the controller and sensor were twisted for the magnetic fields emanating from the cables to cancel out each other.

Finally some changes were made to the thresholds in order to increase sensitivity while eliminating false positives. For example, the reading interval was increased to 40ms (other values adjusted) and the waiting threshold between a locked wheel and the monitor sending an alarm was halved to 0.25s. In order for the change to have any negative effects the scooter would have to ride at the threshold pace and be unfortunate with the alignment. Therefore this was considered a worthwhile optimization for the prototype at hand.

All in all the fine tuning had a considerable impact on the usefulness of the prototype by increasing sensitivity while removing some of the misfires caused by false positives. The entire code content can be found in the *Appendix.*

Electronic configuration

In figure 9 the hardware wiring is shown. It consists of the hall effects sensors connected to the breadboard passing through the resistors before entering the analog input in the Arduino microcontroller. The buzzer and LED are also shown connected to the digital pins of the controller so they can be activated when a skid/burnout is detected.

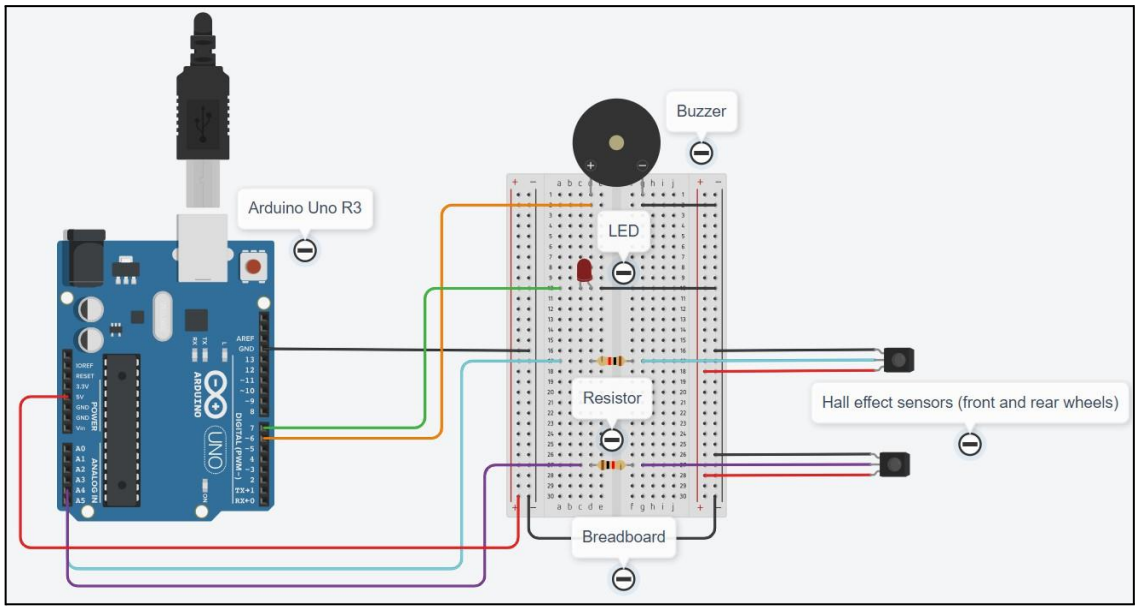


Figure 9. Schematic overview of hardware wiring.

Results

The resulting application monitors the wheels' behavior in order to detect skidding or a burnout. We conducted multiple tests on the developed application in order to assess its reliability.

One important factor for the reliability of the application is its vulnerability to false positive signals. A false positive signal occurs when the application detects a skid or a burnout without a skid or a burnout actually happening. Monitoring the sensor values it could be detected that the signals received by the Arduino do not always match the positioning of the magnet which leads to the false positive signals occurring. In order to detect the probability of false positives occurring, the scooter was used for three minutes of riding at different speed levels and braking as well as accelerating without skidding or causing a burnout. Additionally, the scooter was observed in standstill for two minutes.

Table 1. Test results of false positive test.

application	**false positives during three minute ride**	**false positives during two minutes standstill**
skid detection	2	0

burnout detection	0	2

Additionally, it was tested how well the system detects occurring skids. For this, the scooter was accelerated to maximum speed ten times and then braked to standstill by braking until the rear wheel blocks. The front brake was not used. For the burnout detection, the scooter was lifted up and the back wheel was accelerated for three seconds. This was repeated ten times.

Table 2. Test results of detection quality.

application	**detected skids/burnouts**
skid detection	7 out of 10
burnout detection	10 out of 10

Discussion

The anti abuse system proved to detect skidding and burnouts at a significant level, however the “false positives” needs to be significantly reduced, in order for the system to have a real world application. These “false positives” can be problematic because the incorrect detection of skidding or a burnout can decrease the acceptance of the system. When the system starts giving auditory and visual signals despite no apparent abuse of the scooter, the rider can get the impression that the system is broken and not take the information the system provides him/her with seriously. Also, it is imaginable to report users to the e-scooter company if they frequently display abusive behavior. In that case, the “false positives” would lead to incorrect reporting of users who do not actually abuse the scooters. This would probably decrease user satisfaction with the company and therefore lead to loss in revenue.

All in all the system seems to be very sensitive to skids and burnouts occurring. This sensitivity can be regarded as sufficient for this application.

The underlying causes of the false positives have not yet been fully identified. During the development it was clear that the input was sensitive to impacts, when moving around the number of false positives increased. The most interesting aspect was that there seemed to be a difference between the two sensors, with the rear sensor putting out more false positives than the front. The main parts distinguishing these are the length of the wires and the position of the scooter engine. When testing the sensors we don’t remember identifying any false positives. The wires are likely connected to the issues and the longer wiring to the rear could increase the risk for false signals. Lastly the electric engine is placed at the rear. It is very likely that the magnetic field emitting from the engine could interfere with our sensors. Additionally this is emphasized by the manual of the sensors which mentions that stray magnetic fields might interfere with operations.[8]

To further improve the system, better sensors, wiring and filtering techniques could be used. The system could also utilize the built in speed sensors in the rear wheel, which would probably be very accurate. The system could also be improved to detect skidding faster and stop it in real time. The system would then be considered an ABS system. The detection of a skid in real time is probably reasonable as it would probably still rely on hall effect sensors,

with more magnets/targets (“ABS ring”). The issue would be to stop it in real time as it would require an expensive and complicated Abs unit. It would also require hydraulic brakes. We therefore consider the anti skid system that this project created, a cost effective and simple way to increase safety and compliance of E-scooter riders.

In a real implementation of the system, the user could be notified in their phone if skidding occurs during the ride. If the rider is new to the service, it can be reasonable to assume that the skidding was unintentional. In that case the user could receive some kind of guide or video instruction to improve future riding. If the rider skids very frequently, the skidding could be deemed intentional and the user could first be warned and later possibly banned from the service.

With the test results obtained so far, it is not possible to determine whether the system’s sensitivity of skid/burnout detection varies with the duration of the skid/burnout or the speed at which the skid/burnout occurs. In order to conduct standardized tests on this, it is required to precisely control the speed at which the skid/burnout is started which was not possible for us.

Conclusions

The project showed us that it is possible to develop a basic system notifying the user of abusive behavior when doing skids or burnouts on an e-scooter without many hardware requirements and with a simple software solution. However, the project also made it clear that in order to accurately determine skidding or burnouts, having reliable data from the sensors to process is vital.

References

[1] SVD. (2019). *4 miljoner resor med elsparkcyklar på 8 månader* [Online]. Available: https://www.svd.se/a/e80Abl/4-miljoner-resor-med-elsparkcyklar-pa-8-manader

[2] Folksam. (2020). "Kartläggning av olyckor med elsparkcyklar och hur olyckorna kan förhindra" [Online], Available: https://nyhetsrum.folksam.se/sv/files/2020/08/Folksam-Rapport-Elsparkcyklar-aug-2020-1.pdf

[3] R.R. Pai, M. Dozza. "E-Scooters: Transport or leisure? Findings from naturalistic data collection" in *11th International Cycling Safety Conference*, The Hague, Netherlands, 2023, pp. 16-18

[4] Bird. (2021). *Bird's New Skid Detection Elevates E-Scooter Safety, Rider Compliance* [Online].Available: https://www.bird.co/blog/bird-skid-detection-elevates-e-scooter-safety-rider-compliance/

[5] P. Wallgren (2024). Acceptance (TRA405). Lecture Slides. p. 31

[6] RS Components. *The Guide to Hall Effect Sensors* [Online]. Available: https://se.rs-online.com/web/generalDisplay.html?id=ideas-and-advice/hall-effect-sensors-guide

[7] Arduino. *UNO R3* [Online]. Available: https://docs.arduino.cc/hardware/uno-rev3/

[8] RS Components. *Honeywell Through Hole Hall Effect Sensor, U-Pack, 3-pin* [Online]. Available: https://se.rs-online.com/web/p/motion-sensor-ics/3683945?gb=s

Statement of contribution

Everyone of the group contributed equally with their individual skills to the success of the project. Everybody was present during project work time and report writing. During that time we focused on different things, but discussed and collaborated on all things.

Lukas focused on the programming of the system and the writing of the report. He also conducted the tests of the system.

Mahin participated in designing and producing the mechanical parts of the application and in writing the report. He also created figures and graphics for the project demo presentation.

Jakob participated in designing and producing the mechanical parts of the application, in assembling the electrical parts and in writing the report.

Robin worked on the programming of the system, the assembling of the electrical parts and the writing on the report.

Appendix 1: Code anti-skidding program

Note! Two different applications were made, one for skidding, one for burnout. These are easily integrated into one. The reason for the separation was to limit the effects of false positives.

```
//Include the necessary header file
#include <Arduino.h>

unsigned int CurrentValueRear;
unsigned int LastValueRear;
unsigned int CurrentValueFront;
unsigned int LastValueFront;
unsigned int skidCounter;
int iFront;
int iRear;
int skidTime;
float Speed;
bool  RearWheelLocked;
bool FrontWheelLocked;
int signalThreshold;
int intervalThreshold;
int lockThreshold;
int skidThreshold;
int LED;


void setup() {
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin(9600);
  iRear = 0;
  iFront = 0;
  LastValueRear = analogRead(A5);
  LastValueFront = analogRead(A4);
```

```
  RearWheelLocked = false;
  FrontWheelLocked = false;
  signalThreshold = 10;
  intervalThreshold = 4; // 4*20ms = 80ms
  lockThreshold = 12;
  skidThreshold = 6;
  skidTime = 0;
  skidCounter = 0;
  LED = 1;
}

void loop(){
  rearWheel();
  frontWheel();
  if (RearWheelLocked && FrontWheelLocked){
    //Serial.println("Both wheels locked!");
    skidTime = 0;
  }
  else {
    skidMonitor();
  }
  delay(40);
  //Serial.println(skidTime);
  skidCounter = skidCounter - 1;
  if (skidCounter == 0){
  digitalWrite(7, LOW);
  digitalWrite(8, LOW);
  }
}


void rearWheel(){
  CurrentValueRear = analogRead(A5);
```

```
  if(CurrentValueRear == 0
      && LastValueRear >= signalThreshold
    && iRear > intervalThreshold) {
    Serial.print("rear positive i: ");
    Serial.println(iRear);
    RearWheelLocked = false;
    iRear = 0;
  }
   else if (iRear > lockThreshold) { //24 because 25*20ms is half a second -> blocking it for
more than half a second means skidding
      RearWheelLocked = true;
      }

  else {
    iRear++;
  }
  LastValueRear = CurrentValueRear;
}

void frontWheel(){
  CurrentValueFront = analogRead(A4);
  if(CurrentValueFront == 0
      && LastValueFront >= signalThreshold
    && iFront > intervalThreshold) {
    Serial.print("front positive i: ");
    Serial.println(iFront);
    FrontWheelLocked = false;
    iFront = 0;
  }
   else if (iFront > lockThreshold) { //24 because 25*20ms is half a second -> blocking it for
more than half a second means skidding
      FrontWheelLocked = true;}

  else {
```

```
    iFront++;
  }
  LastValueFront = CurrentValueFront;
}

void skidMonitor(){
  if (RearWheelLocked) {
    if (skidTime > skidThreshold) {
      Serial.println("Skidding! Stop it!");
      LED = LED ^ 1;
      Serial.println(LED);
      digitalWrite(7, LED);
      digitalWrite(8, 1);
      skidCounter = 12;
      // buzzer
      // light
    } else {
      skidTime++;
    }
  } else {
    skidTime = 0;
  }
}
```

Appendix 2: Code anti-burnout program

```
 //Include the necessary header file
#include <Arduino.h>

unsigned int CurrentValueRear;
unsigned int LastValueRear;
unsigned int CurrentValueFront;
unsigned int LastValueFront;
unsigned int skidCounter;
unsigned int burnoutCounter;
int iFront;
int iRear;
int burnoutTime;
float Speed;
bool  RearWheelLocked;
bool FrontWheelLocked;
int signalThreshold;
int intervalThreshold;
int lockThreshold;
int burnoutThreshold;
int LED;


void setup() {
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  Serial.begin(9600);
  iRear = 0;
  iFront = 0;
  LastValueRear = analogRead(A5);
  LastValueFront = analogRead(A4);
```

```
  RearWheelLocked = false;
  FrontWheelLocked = false;
  signalThreshold = 10;
  intervalThreshold = 4; // 4*20ms = 80ms
  lockThreshold = 12;
  burnoutThreshold = 18;
  burnoutTime = 0;
  burnoutCounter = 0;
  LED = 1;
}

void loop(){
  rearWheel();
  frontWheel();
  if (RearWheelLocked && FrontWheelLocked){
    //Serial.println("Both wheels locked!");
    burnoutTime = 0;
  }
  else {
    burnoutMonitor(); // Uncomment if you want to monitor burnout as well
  }
  delay(40);
  burnoutCounter = burnoutCounter - 1;
  if (burnoutCounter == 0){
  digitalWrite(7, LOW);
    digitalWrite(8, LOW);
  }
}



void rearWheel(){
```

```
  CurrentValueRear = analogRead(A5);
  if(CurrentValueRear == 0
      && LastValueRear >= signalThreshold
    && iRear > intervalThreshold) {
    Serial.print("rear positive i: ");
    Serial.println(iRear);
    RearWheelLocked = false;
    iRear = 0;
  }
   else if (iRear > lockThreshold) { //24 because 25*20ms is half a second -> blocking it for more than half a second means skidding
      RearWheelLocked = true;
      }

  else {
    iRear++;
  }
  LastValueRear = CurrentValueRear;
}

void frontWheel(){
  CurrentValueFront = analogRead(A4);
  if(CurrentValueFront == 0
      && LastValueFront >= signalThreshold
    && iFront > intervalThreshold) {
    Serial.print("front positive i: ");
    Serial.println(iFront);
    FrontWheelLocked = false;
    iFront = 0;
  }
   else if (iFront > lockThreshold) { //24 because 25*20ms is half a second -> blocking it for more than half a second means skidding
```

```
      FrontWheelLocked = true;}

  else {
    iFront++;
  }
  LastValueFront = CurrentValueFront;
}



void burnoutMonitor(){
  if (FrontWheelLocked) {
    if (burnoutTime > burnoutThreshold) {
      Serial.println("Burnout! Stop it!");
      LED = LED ^ 1;
      Serial.println(LED);
      digitalWrite(7, LED);
      digitalWrite(8, 1);
      burnoutCounter = 12;
      // buzzer
      // light
    } else {
      burnoutTime++;
    }
  } else {
    burnoutTime = 0;
  }
}
```