

RoadSense

Real-Time Road Condition Detection
and Campus Pothole Mapping

ECE 445 Project Proposal · Fall 2026

D'Angelo Manzo (dmanz2)

Adam Moine (moine2)

Jovan Kular (hkula2)

TA: Shengkun Cui

1. Introduction

1.1 Problem

Roads around campus are in bad shape, and nobody measures them.

A pothole is an annoyance in a car. It is a crash on a bike. Car wheels are big enough to roll over most defects. A bicycle or scooter wheel is small enough to drop into one, and then the rider goes over the handlebars. Thousands of students at UIUC bike or scooter to class every day. Many of them ride in bike lanes that nobody has ever surveyed.

The money side is large too. TRIP, a transportation research group, found that rough urban roads cost U.S. motorists about \$130 billion a year in extra wear, fuel, and repairs, which works out to about \$1,049 per driver. A 2016 AAA study put pothole damage alone at roughly \$3 billion a year, spread across 16 million drivers at an average of \$300 per repair. The American Society of Civil Engineers gave U.S. roads a **D+** in its 2025 Report Card and reports that 39% of major roads are in poor or mediocre condition. These costs hit hardest on people who cannot absorb a \$400 bent rim.

Cyclists carry more of the risk and less of the protection. NHTSA recorded 1,166 bicyclists killed in crashes with motor vehicles in 2023. Surface defects are a separate hazard on top of that, and one nobody counts, because a rider thrown by a pothole with no car involved usually does not appear in any crash statistic at all.

Cities cannot fix what they cannot see. There are three ways road condition gets measured today, and all three have gaps.

1. **A person drives around and scores the road by eye.** Slow, subjective, and two raters often disagree.
2. **A survey truck measures the road properly.** These are specialized vehicles carrying laser sensors, and running a program with them is expensive. The FHWA estimates about \$6.3 million to set up roadway data collection for an average state, plus \$3.4 million to keep it going. At that price the trucks get sent to busy main roads, because that is where the budget can be justified. Side streets get skipped every cycle. Bike lanes are not included at all.
3. **People call 311.** You get "big pothole on Green Street." No exact location, no depth, and no way to tell whether it is worse than the other fifty open reports.

So a city cannot decide what to fix first, and a rider cannot find out whether a route is safe before taking it.

This is more than an inconvenience, and it reaches into most of the areas engineers are asked to weigh.

Public health, safety and welfare. Cyclists and scooter riders have the least protection and are hurt worst by bad pavement. Injuries from a thrown rider land in emergency rooms, and the fear of a bad route keeps people in cars who would rather not be.

Economic. Road damage gets worse quickly once it starts. Sealing a crack early costs a small fraction of rebuilding the road later, so measuring sooner saves money that is otherwise spent on repairs no one planned for.

Environmental. Rough pavement makes vehicles burn more fuel, and rebuilding a road takes a lot of concrete, asphalt, and diesel that catching the damage early would avoid.

Social and cultural. People choose how to travel partly on whether it feels safe. Streets that are unpleasant to ride push a community back towards driving, which shapes how a place feels to live in. The streets nobody measures are usually the ones already being neglected, and the missing data becomes the reason nothing changes.

Global. Poor road data is not a US problem. Most low- and middle-income countries have no systematic road condition surveys at all, because the equipment costs more than the roads budget. A measuring tool cheap enough to be everywhere is worth more there than here.

The problem we are solving: riders and drivers have no real information about the condition of the streets they actually use, and the tools that could produce that information cost more than any campus or neighborhood can afford.

1.2 Solution

We are building a small sensor box that mounts on a car or a bicycle. While you drive, it measures how the vehicle moves, decides on the spot whether each jolt was a pothole, a speed bump, or just rough pavement, and saves that decision together with the GPS location. **All of the detecting and deciding happens on the device while you are driving.** It does not store raw data for a computer to sort out later. After the trip, the memory card goes into a laptop, which adds the new events to a campus database and redraws a map of UIUC roads and bike lanes colored by condition. Once enough trips are recorded, that map supports the feature we actually want: give it a starting point and a destination on campus, and it returns a route that avoids the worst pavement.

The deciding works on one physical idea: **a speed bump shakes you slowly, a pothole shakes you fast.** A speed bump is long and smooth, about 3.7 m, so crossing it takes about half a second and the car body rises and falls roughly one to three times per second. A pothole is short and sharp; the tire slams into its far edge in about 11 milliseconds, which shakes the car from about 5 to 60 times per second. Those two speeds of shaking are far enough apart to measure and tell apart, and two more clues help: a pothole pushes the wheel **down** first while a bump pushes it **up**, and the length of the jolt multiplied by your speed tells you how physically long the defect was no matter how fast you were going.

Inside the box are four sensors and two small computers:

- A **BMI270** motion sensor measures how fast the box speeds up and slows down in each direction, and how fast it rotates. The rotation part matters: it tells the device how it is tilted, so it can tell real up-and-down motion apart from braking or turning.
- An **analog accelerometer** with a custom filter circuit catches the fast, hard impacts and the high-frequency buzz of rough pavement, both too quick for the BMI270.

- A **MAX-M10S GPS receiver** gives location and speed. Speed matters, because the same pothole produces a much bigger jolt at 40 km/h than at 20 km/h.
- An **STM32F411 microcontroller** reads every sensor at exactly the right moment, works out the tilt, filters the signals, spots events, and makes the decision using the rule above.
- A **Raspberry Pi** we already own runs a trained model on those events to catch the harder cases the rule gets wrong, and saves the results. The model is trained beforehand on a laptop from data we record; nothing is trained while driving. The rule on the microcontroller runs every time regardless, so the device still works with the Pi switched off.

1.3 Visual Aid

Figure 1 appears on the following page.

Figure 1 — RoadSense in Use

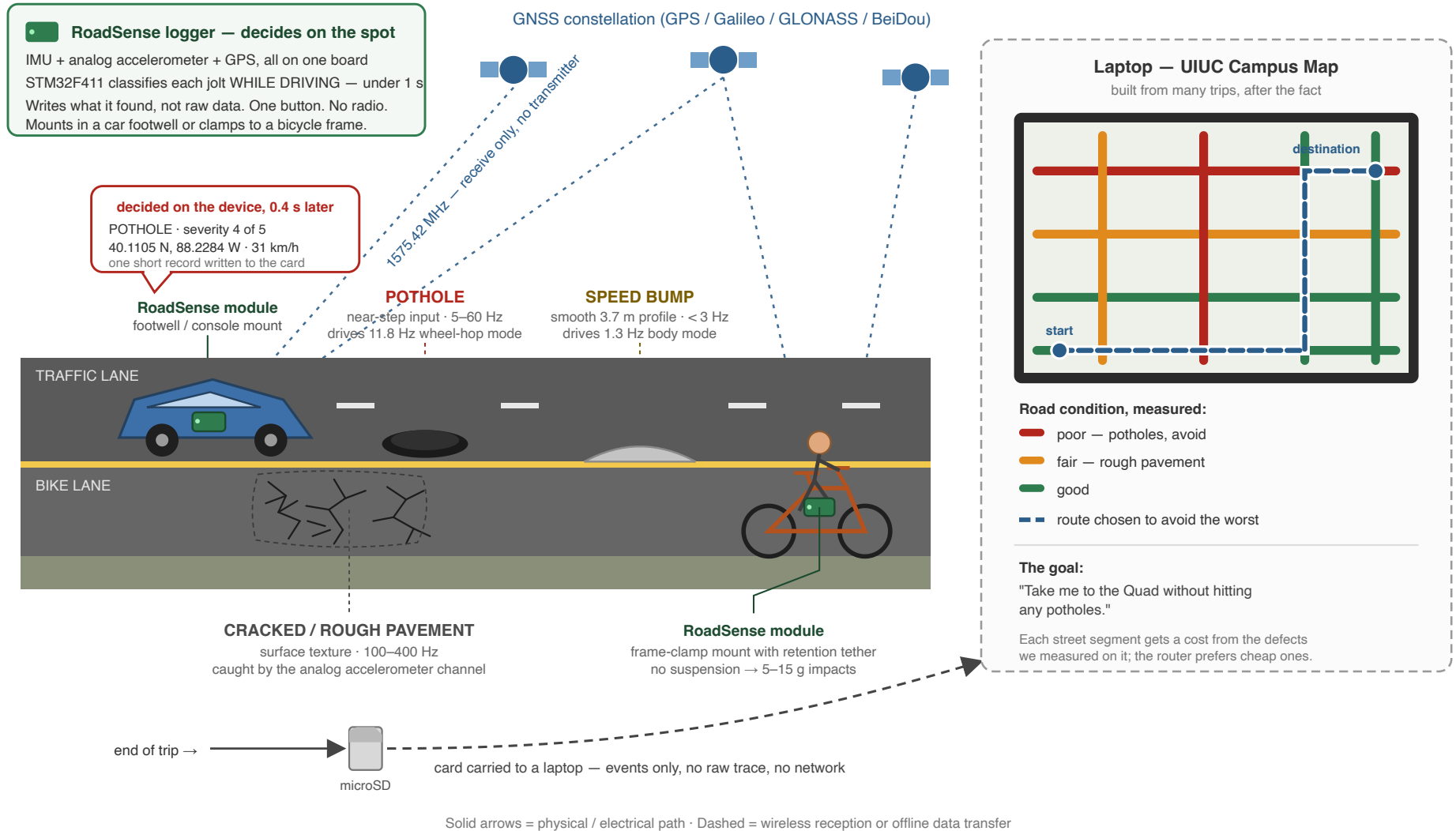


Figure 1. How RoadSense is used. The device mounts on a car or bicycle and decides what each jolt was while you drive. After the trip the card builds a campus condition map that supports pothole-avoiding routing.

1.4 High-Level Requirements

- **HLR-1 (Works in real time).** The device shall take at least 100 motion readings per second and at least 1,000 readings per second from the analog accelerometer. It shall decide what each road event was **on the device, within 1 second of the event happening**. It shall run for at least 30 minutes without losing or damaging any data, which we check using counters and checksums saved in the log.
- **HLR-2 (Gets it right).** Driven over a surveyed campus route with at least 60 potholes and 30 speed bumps, at three different speeds, the device shall find at least 90% of the potholes, shall correctly label at least 80% of the events it finds as either "pothole" or "speed bump," and shall report no more than 1 false event per kilometer of smooth road. This shall be met by the simple rule running on the STM32 on its own, with the trained model on the Raspberry Pi measured separately so the two can be compared.
- **HLR-3 (Knows where, and can route around it).** Every event shall be saved within 10 m of where it really is. The laptop program shall draw a map of the surveyed campus roads colored by condition, and shall find a route between any two campus points that avoids roads worse than a limit the user picks.

2. Design

2.1 Block Diagram and Mechanical Drawing

There are seven subsystems: Power, Motion Sensing, Analog Front End, Location, Acquisition and Timing, Model and Storage, and Mapping and Routing. The last one runs on a laptop after the trip, because a map is built from many trips and cannot be made during one.

Figure 2 appears on the following page.

Figure 2 — RoadSense Block Diagram

Red = power, with voltages · Black = data, with the protocol named · Grey dashed = mechanical, radio, or card carried by hand

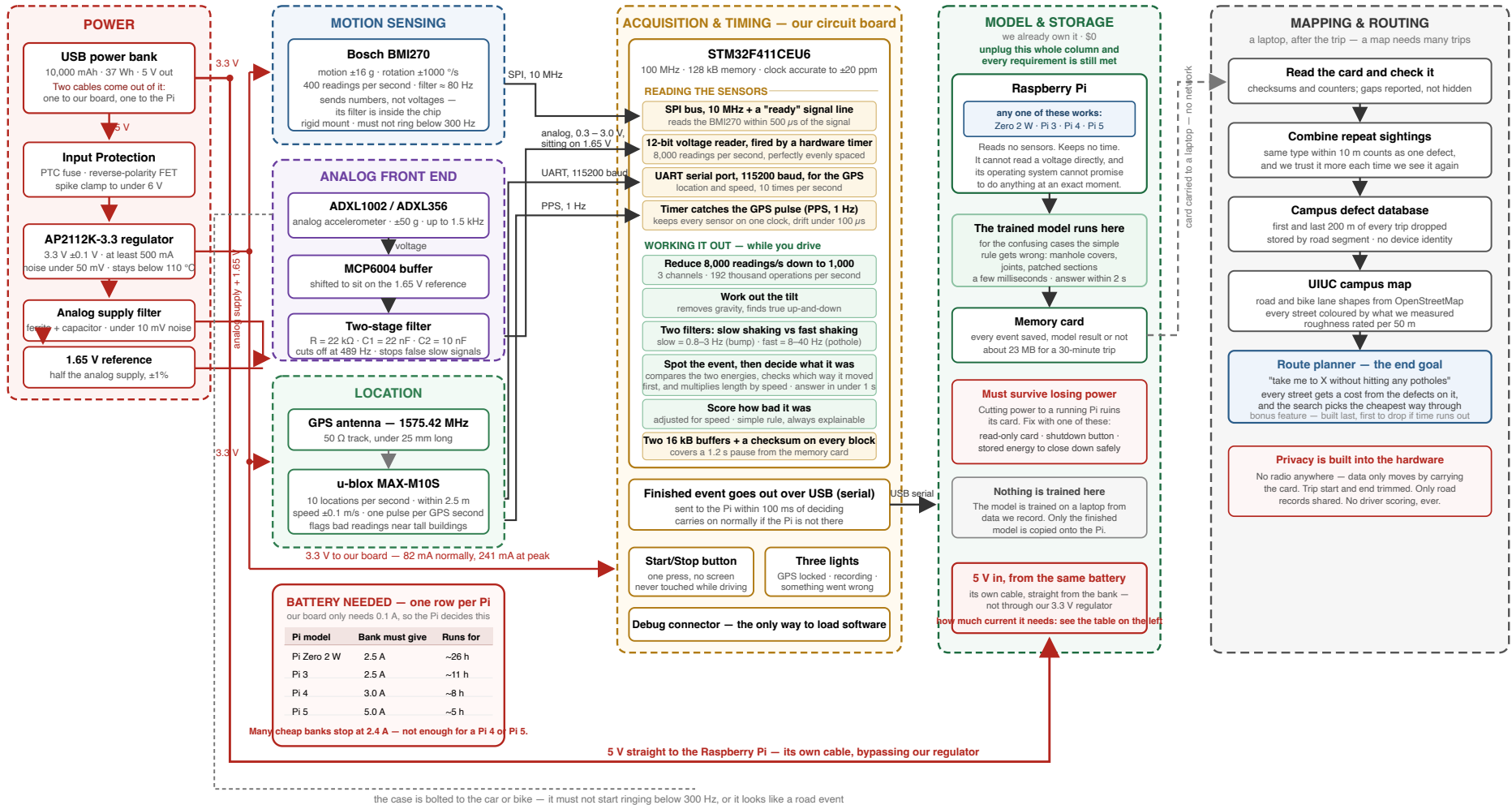


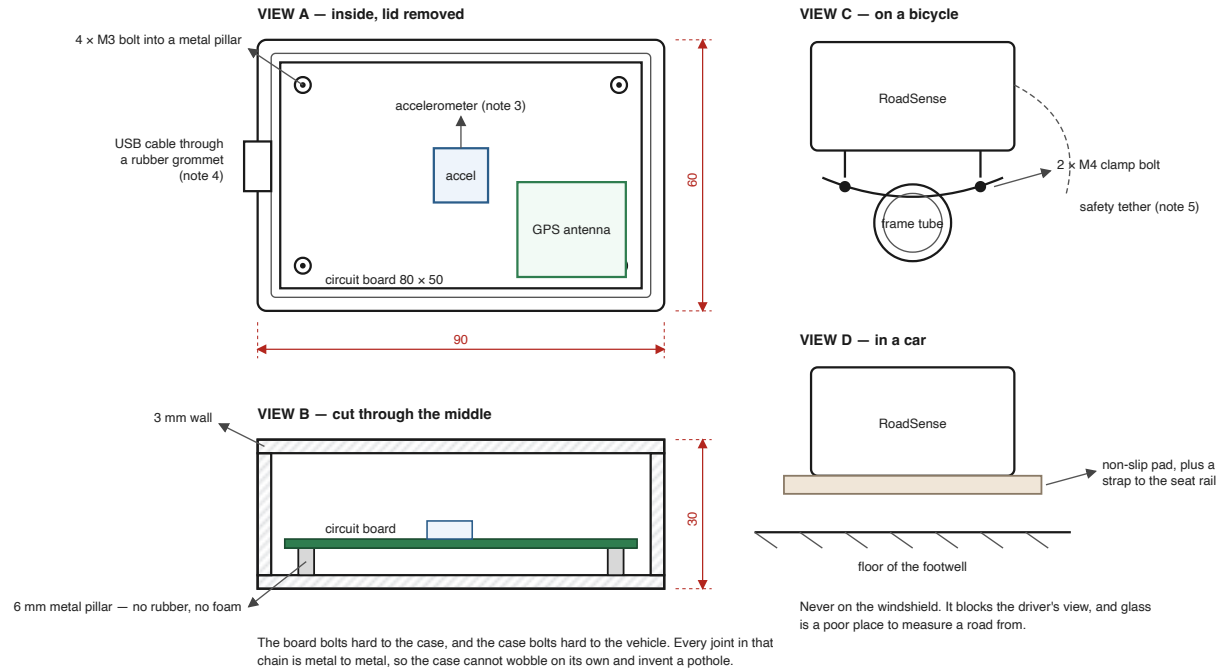
Figure 2. Block diagram. Seven subsystems. All timing-critical work happens on our own circuit board; the Raspberry Pi only improves the hard cases and can be removed entirely.

The case and how it mounts. The case matters more than it looks. It is the mechanical link between the road and the sensor, so any flex or rattle in it arrives in the data looking like a pothole that was never there.

Figure 3 appears on the following page.

Figure 3 — Enclosure and Mounting

All dimensions in millimetres · views are not to the same scale



Red = dimension in millimetres · Hatching = material the section view cuts through

Notes

1. The case is part of the sensor.

Anything that can flex or rattle between the board and the vehicle shows up in the data looking like a bump in the road. The whole assembly must not ring below 300 times a second. We test that by tapping it and watching what the sensor reports.

2. Metal pillars, not rubber feet.

Rubber and foam are exactly what we have to avoid here. They soften the very shaking we are trying to measure.

3. The accelerometer sits in the middle.

Placed at the centre of the four bolts, where the board flexes least. Near an edge it would measure the board bending rather than the vehicle moving.

4. Every cable is supported where it enters.

A cable that can swing will eventually break off at its solder joint. This device exists to be shaken, so that is a question of when, not whether.

5. The bike mount has a second line of defence.

A clamp can loosen over a few hundred bumps. The tether means that if it does, the case dangles instead of going into the spokes.

6. The case opens.

A lid on four screws, a memory card you can reach, and a battery on a plug. Nothing is glued or sealed shut, so it can be repaired rather than thrown away.

7. Same case either way.

One enclosure, two brackets. Nothing inside changes between the car and the bicycle, so both share one calibration of the mounting.

Figure 3. The case and how it mounts. Every joint between the board and the vehicle is metal to metal, so the case cannot wobble on its own and register a road event that did not happen.

2.2 Subsystem Overview

Power. A USB power bank feeds the whole system through **two separate cables**. One goes to our circuit board, where protection parts stop surges and reversed connections, and an AP2112K-3.3 regulator makes the single 3.3 V supply the sensors and microcontroller run on. The other goes straight to the Raspberry Pi at 5 V, bypassing our regulator entirely, because the Pi needs far more current than our regulator can provide. A separately filtered supply and a steady 1.65 V reference go to the Analog Front End, so that electrical noise from the memory card does not leak into the measurements.

Motion Sensing. A BMI270 sensor soldered to the main board. It sends motion readings to the microcontroller over a wire pair called SPI, and raises a separate signal line every time a new reading is ready. That way the microcontroller reads on the sensor's schedule instead of guessing when data is available. This sensor sends numbers, not voltages, so its filtering is built into the chip and set by writing to a register. There is no external filter circuit for it.

Analog Front End. An analog accelerometer sends out a voltage instead of a number, so we have to build everything that comes after it. Each axis gets buffered, then passed through a filter built from an MCP6004 amplifier chip, then read by the microcontroller. The filter is there to stop a problem called aliasing, explained in the requirements below.

Location. A MAX-M10S GPS receiver with a small flat antenna. It sends location, speed, and direction to the microcontroller ten times a second over a serial connection. It also sends a separate electrical pulse once every GPS second. That pulse is what keeps all the sensors on one shared clock.

Acquisition and Timing. The STM32F411 microcontroller, on our own circuit board. It reads all three sensors, stamps everything with a time taken from the GPS pulse, and does the time-critical work while driving: filtering, working out tilt, spotting events, and making a first decision using a simple rule. It sends each finished event to the Raspberry Pi over a USB cable. Three LEDs show GPS lock, recording, and errors. One button starts and stops a trip.

This subsystem holds the timing-critical work on purpose. A Raspberry Pi cannot do this job: it has no way to read a voltage directly, and the operating system it runs cannot promise to do anything at an exact moment.

Model and Storage. A Raspberry Pi we already own. It receives events from the microcontroller, runs a trained model on each one to sort out the cases the simple rule finds confusing, and saves the results to a memory card. It does no sensor reading and no timing. If it is switched off or fails, the microcontroller keeps working and keeps its own simple decisions, so a trip is never lost.

Mapping and Routing. A Python program on a laptop. It reads the card, adds events to a database, draws the campus map, and finds routes. This part is deliberately not on the device: it needs data from many trips, so it cannot run during one.

2.3 Subsystem Requirements

Each requirement below is something we have to achieve and can measure ourselves. We have deliberately not written down numbers that a chip's datasheet already promises, because restating a manufacturer's specification is not a requirement on our design.

Power

How it serves the high-level requirements: every other subsystem stops if the supply sags. A dip while writing to the memory card damages the record being written, which breaks HLR-1.

1. Hold $3.3\text{ V} \pm 0.1\text{ V}$ at the microcontroller's own power pin for a whole 30-minute trip, measured with a scope at the pin rather than at the regulator.
 2. Keep electrical noise under 50 mV peak-to-peak at the digital chips and under 10 mV at the analog chips, measured at their own pins while the memory card is being written.
 3. Hold the rail within $\pm 100\text{ mV}$ when the current suddenly jumps from 90 mA to 260 mA, tested with a switched load.
 4. Keep the outside of the regulator below $90\text{ }^{\circ}\text{C}$ after 30 minutes of continuous running in a $60\text{ }^{\circ}\text{C}$ space, measured with a thermocouple. This is a test of our board layout and copper area, not of the regulator itself.
 5. Survive 5 V connected backwards, and a 10 V spike on the input, with no part of the board damaged afterwards.
 6. Keep the analog reference at half the analog supply within $\pm 1\%$, and keep it from moving more than 1 mV while the analog signal swings across its full range.
 7. Feed the Raspberry Pi on a path separate from our own 3.3 V supply, so that our rail does not move when the Pi draws current. Our regulator must never be asked to feed the Pi.
- How much the battery has to give depends on which Pi we have, as Table 1 shows.

Table 1. What the battery must supply, for each Raspberry Pi we might have.

If the Pi is a...	Power it uses	Battery must supply	Battery lasts
Pi Zero 2 W	about 1 W	2.5 A	about 26 hours
Pi 3	about 3 W	2.5 A	about 11 hours
Pi 4	about 4 W	3.0 A	about 8 hours
Pi 5	about 7 W	5.0 A	about 5 hours

Our own board adds 0.1 A on top of whichever row applies. Many cheap battery packs stop at 2.4 A, which is not enough for a Pi 4 or Pi 5, so we test the pack under load.

Motion Sensing

How it serves the high-level requirements: the rotation reading is what makes the up-and-down measurement trustworthy. Without it, a box mounted at an angle turns braking into a false pothole. The reading rate decides whether the fast shaking that identifies a pothole survives at all, which is HLR-2.

1. Deliver acceleration and rotation to the microcontroller at 400 sets per second or faster, losing fewer than 1 set in 10,000 across a 30-minute run, counted from the sequence numbers in the log.
2. Not run off the top of the scale on the impacts we actually need to record: up to 15 g up-and-down on a bicycle and up to 5 g in a car.
3. Let the software work out which way is down to within 5°, with the case mounted at any angle up to 30° off level. Tested by fixing the case at known angles and comparing.
4. Keep the whole subsystem's background noise at least 40 times smaller than the smallest bump we must catch, which is 0.2 g. Tested by logging while the vehicle sits still.
5. Be mounted so that the board and case together do not ring below 300 Hz, tested by tapping the assembly and looking at what comes back. A ringing case would look like a road event.

Analog Front End

How it serves the high-level requirements: this covers the fast impacts and the fine buzz from 100 to 400 times per second that separates cracked pavement from smooth. The BMI270 cannot reach that range, so without this subsystem the rough-versus-smooth part of HLR-2 fails.

1. Pass shaking between 16 and 400 times per second, with each cutoff within $\pm 20\%$ of where we aim it, measured by feeding a swept signal into the circuit and recording what comes out.
2. Shrink anything shaking at 2,000 times per second down to a twenty-fifth of its size or less.
Why this matters: the microcontroller takes readings at a fixed rate. Anything shaking faster than half that rate does not just get missed. It gets **mistaken for a slow signal** and shows up in the data as something that never happened. This is called aliasing, and no amount of software can remove it afterwards. The filter has to take those fast signals out before the reading is taken.
3. Keep its output between 0.3 V and 3.0 V for impacts up to ± 30 g, so nothing is clipped off at either end.
4. Survive the accelerometer output being shorted to ground or driven above the supply, without damage to the amplifier.
5. Add less than 5 mV of its own noise, measured with the input held still.
6. Settle to within 1% of its correct value within 100 ms of power-up, so the start of a trip is usable.

Filter parts, one set per axis: two 22 k Ω resistors, one 22 nF capacitor, one 10 nF capacitor.

Working out where it cuts off:

$$\text{cutoff} = 1 / (2\pi \times 22,000 \times \sqrt{(22 \text{ nF} \times 10 \text{ nF})}) = 489 \text{ times per second (489 Hz)}$$

Those particular capacitor values also keep the response flat right up to the cutoff, with no bump just before it.

Location

How it serves the high-level requirements: it turns "a jolt happened" into "a jolt happened at this corner," which is all of HLR-3. Its speed reading is what makes severity scores comparable between fast and slow trips. Its timing pulse is what lets us prove the shared clock in HLR-1.

1. Deliver position, speed and time to the microcontroller at 10 sets per second or faster, losing no more than 1 set per minute of driving.
2. Place every detected event within 10 m of where it really is, checked against our surveyed campus route. This is the requirement that HLR-3 is graded on.
3. Mark any reading the receiver reports as poor, so the microcontroller can work out position from the last known speed and direction instead of saving an event in the wrong place.
This matters on campus, where tall buildings bounce GPS signals and can push the reported position off by 10 to 30 m.
4. Provide a once-per-second pulse that the microcontroller can catch, so that every sensor reading in the log shares one clock to within 10 ms.
5. Keep the antenna track on our board at 50 Ω and under 25 mm long, routed away from parts that switch quickly.

Acquisition and Timing

How it serves the high-level requirements: this subsystem is HLR-1. It is also where the deciding in HLR-2 happens, since the requirement is that the device works it out during the trip.

1. Read all three sensors at the same time at their full rates — 400 per second for motion, 8,000 per second for analog, 10 per second for GPS — and drop nothing across 30 minutes.
2. Run the whole processing chain while driving: rate reduction, tilt estimate, the two band filters, event detection, deciding, and severity scoring. Total processor load shall stay under 25%, measured by toggling a pin and watching it on a scope.
3. Produce a decision within 1 second of the event, timed from the first reading that crosses the detection threshold.
4. Hold at least 16 kB in each of two buffers, giving 800 ms or more of slack when the memory card pauses. If a buffer ever overflows it shall be counted, written to the log, and shown on an LED. Never silently dropped.
5. Tag every block written with a counter and a checksum, so the "nothing lost" claim in HLR-1 can be checked afterwards rather than assumed.
6. Keep its own clock locked to the GPS pulse so that it drifts less than 100 μ s between pulses, measured by comparing captured pulse times across a whole trip.
7. Draw no more than 260 mA at peak and 120 mA on average from the 3.3 V supply, so the power subsystem stays inside its own limits.

8. Close the log and flush everything within 500 ms of the stop button being pressed, so pulling the power straight after does not lose the trip.
9. Provide a debug connector, since there is no other way to load software onto the board.
10. Send each finished event to the Raspberry Pi within 100 ms of deciding, and keep working normally if the Pi is absent, switched off, or not responding.

Model and Storage (Raspberry Pi)

How it serves the high-level requirements: it improves the labelling accuracy in HLR-2 on the events the simple rule finds confusing. It is deliberately kept out of HLR-1, because nothing time-critical happens here and the device has to meet HLR-1 without it.

We already own a Raspberry Pi. We have not checked which model it is yet, and Table 1 lists every one it could be. Whichever it turns out to be, the design works. The only things that change are how long the battery lasts and how much current it has to give, and even the worst row in that table beats the 30 minutes we need by ten times.

1. Receive events from the microcontroller over USB and produce a model result for each one within 2 seconds. This is deliberately looser than the 1-second requirement in HLR-1, because HLR-1 is met by the microcontroller on its own.
2. Save every event to a memory card, whether or not the model succeeded, so no data is lost if the model crashes.
3. Survive having its power cut without damaging its memory card. We will do this by making the card read-only, or by adding a shutdown button, or by adding enough stored energy for it to close down on its own. **Pulling the plug on a running Raspberry Pi corrupts its card, and this is the most common way a project like ours fails on demo day.**
4. Start up and be ready to receive events within 60 seconds of power-on.
5. Run the model in under 100 ms per event. A small model over a window of readings is roughly 50,000 to 200,000 multiply-add operations, which any Raspberry Pi finishes in a few milliseconds. There is a large amount of room here.
6. Be entirely optional. The device shall meet every high-level requirement with the Pi removed.

Mapping and Routing

How it serves the high-level requirements: it produces the campus map and the route, which together are all of HLR-3.

1. Read the event records and check every counter and checksum, reporting gaps rather than hiding them.
2. Combine repeat sightings of the same defect across trips. Two events of the same type within 10 m are treated as one defect, and confidence goes up each time it is seen again.
3. Draw campus roads and bike lanes colored by measured condition, using road shapes from OpenStreetMap.

4. Find a route between two campus points that avoids roads worse than a limit the user picks. Each road segment gets a cost that goes up with how many defects are on it and how bad they are, and the route search prefers cheap segments.
5. Process a 30-minute trip in under 60 seconds on a normal laptop.

2.4 Tolerance Analysis

The biggest risk to this project is simple. If potholes and speed bumps do not actually shake the car differently, nothing else matters, because no amount of software can tell apart two things that look the same. So that is what we check here.

To answer this we use the standard simple model of one corner of a car (Gillespie, *Fundamentals of Vehicle Dynamics*). It treats the car as two masses connected by springs: the **body** sitting on the suspension, and the **wheel and hub** sitting on the tire. Table 2 gives the values we use, which are typical for a mid-size car.

Table 2. Quarter-car model values used in this analysis.

What it is	Value
Body mass over one wheel	300 kg
Wheel and hub mass	40 kg
Suspension spring stiffness	20,000 N/m
Suspension shock absorber	1,500 N·s/m
Tire stiffness	200,000 N/m

Each mass bounces at its own natural speed. Working those out:

The body bounces at:

$$\text{frequency} = (1/2\pi) \cdot \sqrt{(20,000 / 300)} = 1.30 \text{ times per second (1.30 Hz)}$$

The wheel bounces at:

$$\text{frequency} = (1/2\pi) \cdot \sqrt{(220,000 / 40)} = 11.8 \text{ times per second (11.8 Hz)}$$

These are 9 times apart. Both keep ringing for a few cycles before dying out. So the car behaves like two separate channels: a slow one and a fast one. The question is which channel each event uses.

A speed bump. A standard speed hump is 3.7 m long and meant to be crossed at 15 mph (6.7 m/s). The car is on it for

$$3.7 \text{ m} \div 6.7 \text{ m/s} = 0.55 \text{ seconds}$$

A push that lasts 0.55 seconds contains almost nothing faster than about 2 times per second. Even at 25 mph it stays under 3. That lands right on the body's 1.30 Hz bounce, so the body

rocks strongly. The wheel's 11.8 Hz channel gets almost nothing.

A pothole. Take a hole 0.5 m long and 6 cm deep, hit at 30 mph (13.4 m/s). The car is over it for

$$0.5 \text{ m} \div 13.4 \text{ m/s} = 0.037 \text{ seconds}$$

In that time the tire only falls

$$\frac{1}{2} \times 9.81 \text{ m/s}^2 \times (0.037 \text{ s})^2 = 6.7 \text{ mm}$$

out of 6 cm. **So the tire never reaches the bottom.** What actually happens is that the tire slams into the far wall. How fast that impact is depends on how quickly the contact patch climbs the edge:

$$0.15 \text{ m} \div 13.4 \text{ m/s} = 0.011 \text{ seconds}$$

An 11 millisecond impact contains shaking up to roughly

$$0.35 \div 0.011 \text{ s} = 32 \text{ times per second, with real content out to about } 60$$

That covers the wheel's 11.8 Hz bounce and the tire's own 40–80 Hz ring. Both get hit hard.

So here is the test. Measure how much shaking energy lands in the slow band and how much lands in the fast band, then divide:

$$R = (\text{energy from } 0.8 \text{ to } 3 \text{ Hz}) \div (\text{energy from } 8 \text{ to } 40 \text{ Hz})$$

Table 3 gives what we expect that ratio to be for each kind of event.

Table 3. Expected energy ratio R, worked out from the model above.

Event	Expected R
Speed bump	10 to 30
Pothole	0.05 to 0.3

These are more than 100 times apart. If we put the dividing line at $R = 1$, we are about 10 times away from the nearest edge of either group. That means every number in the table above could be wrong by a factor of 3 and the test would still work. That margin is why the 80% target in HLR-2 is realistic rather than hopeful.

Two more clues back it up: which way the wheel moved first (down for a pothole, up for a bump), and how much fine buzz the analog channel picked up between 100 and 400 Hz, which is large for a sharp edge and near zero for a smooth bump.

What this means for the hardware. We need clean measurements out to 40 Hz, and preferably 60 Hz. Reading only 100 times per second means the fastest thing we could see is 50 Hz, which is inside the range we care about with no room to spare. That is why the BMI270

runs at 400 readings per second with its filter at 80 Hz. We end up sampling the range of interest 10 times more finely than the minimum, which makes aliasing a non-issue.

Bikes are different. A bike has no suspension, so its frame and wheels ring at 15–40 Hz instead of 1.3 and 11.8 Hz. R shifts as a result, so we have to work out separate dividing lines for bikes. We plan to do that rather than assume one number covers both.

3. Ethics, Safety, Standards, and Societal Impact

3.1 Ethics

Location privacy is the main ethical risk. This device records where its owner goes. Location trails are some of the most identifying data there is. De Montjoye et al. (*Scientific Reports*, 2013) showed that just four time-and-place points are enough to uniquely pick out 95% of people in a set of 1.5 million. A record of a student's daily commute gives away their address, their class schedule, and their routine. That is far more than we need to know about pavement.

The IEEE Code of Ethics commits us to respecting privacy, and the ACM Code gives it a full principle. Both are clear that collecting more personal data than the job needs is itself the harm, whether or not it ever leaks.

We handle this by building the privacy in rather than adding it later:

- **The device has no wireless transmitter.** Data only leaves it if the owner physically carries the card somewhere. That is a guarantee built into the hardware, not a promise in a policy document.
- **Trip start and end get trimmed automatically.** The program drops the first and last 200 m of every trail, so a trip does not literally begin at someone's front door.
- **Only road records would ever be shared.** If we add a way to contribute data, it would send the road segment, the type of defect, how bad it is, and a rough timestamp. Never the trail, never a device ID, never an account. Opt in each trip.
- **We will not build driver scoring.** The same data that measures pavement also measures hard braking and cornering. Turning a road monitor into a tool for watching drivers is one small change away. Saying where that line is, out loud, is part of our job. On a campus this is concrete: a device like this could easily be turned into a way to track where students go.

Being honest about what it can do is the second risk. The IEEE Code requires realistic claims. The ACM Code requires thorough testing. Our target is finding 90% of potholes, which means one in ten gets missed. Our labeling target is 80%, which means one event in five gets called wrong. A rider who trusts our map and gets hurt by something we missed was harmed by our overstatement, not just by the pothole.

So we will give ranges instead of single numbers, show how old the data is and how confident we are next to every defect on the map, mark the program as advisory in the interface itself, and never call this a safety system. It is a measuring and mapping tool.

Our data being used against other people. Road data can end up in a lawsuit against the university or the city. That may be fair, but only if the data is right. A badly calibrated severity score, or a GPS error that puts a defect on the wrong block, could support an unfair claim. We will publish how we calibrated and what our error bounds are alongside anything we release, and we will not publish scores for vehicle types we have not tested.

Academic honesty. We will credit every library, reference design, and dataset we use. We will report what we actually measured, including the routes where our program did worst, rather than picking our best runs. The ACM Code specifically forbids that kind of cherry picking.

3.2 Safety

Road testing is the main physical danger. Meeting HLR-2 means deliberately driving and riding over real defects.

- Every vehicle test uses **at least two people**: one driving with full attention, one working the device. Nobody sets up, starts, or reads the device while driving. Illinois law (625 ILCS 5/12-610.2) bans handheld device use while driving, and our one-button interface exists so that nothing ever needs touching in motion.
- **No deliberately hitting potholes above 35 mph**, and nothing deep enough to risk damaging a wheel. We measure defects on foot with a tape measure first, so we know how deep they are before we drive over them.
- **Bike testing** on quiet campus routes, in daylight, helmets on, under 20 km/h over defects.
- The device is **firmly secured**. Loose in a footwell it can roll under a brake pedal. Loose on a bike it can get into the spokes. In a car it sits on a non-slip base or gets strapped to the console. On a bike it uses a clamp plus a backup tether. Nothing goes on the windshield, since blocking the driver's view is restricted under 625 ILCS 5/12-503.

Electrical danger is low. Everything runs at 3.3 V and 5 V and uses under 1.5 W, so there is no shock risk. The real risks are the battery and the car's power:

- Only a **store-bought, UL 2054 / IEC 62133-certified USB power bank**. We will not build a lithium battery pack, will not charge one unattended, and will throw away any pack that swells.
- If we run off a car, we use a certified 12 V-to-USB adapter, so that the electrical spikes cars produce get handled by a device designed for it. Our fuse, reverse-polarity protection, and spike clamp are a second line of defense.

Lab safety follows normal ECE 445 and UIUC Division of Research Safety practice: soldering with fume extraction and eye protection, anti-static straps when handling the motion and GPS chips, and nothing left powered unattended.

3.3 Standards

Six sets of rules apply to this project. For each one we say what it covers and what we actually have to do about it.

FCC Part 15, Subpart B — electronic devices that give off radio noise. Our device has no transmitter. The GPS receiver only listens, so the rules for radio transmitters do not apply to us. What does apply is Subpart B, which covers any digital device that gives off radio noise as a side effect of running a clock. A student project that is not being sold does not need to be certified, but the limits still shape our board: we keep the fast-switching parts away from the GPS antenna track and keep that track under 25 mm. If we ever added wireless, we would need a radio module that is already certified, and the privacy discussion in §3.1 would have to be reopened.

UL 2054 and IEC 62133-2 — portable lithium batteries. These cover how a battery pack must behave when it is overcharged, shorted, crushed, or overheated. We meet them the simple way: we buy a battery pack that is already certified to them and never build one. We also do not charge a pack while it is connected to the device, and we retire any pack that swells.

ISO 2631-1 — how strongly people feel vibration. This standard says which speeds of shaking the human body actually notices, and gives a weighting that reflects that. We use it to turn a raw measurement into a severity score, so that "how bad is this pothole" means the same thing to us as it does to anyone else using the standard. This is why we do not invent our own severity number.

ISO 8608 — describing road surfaces. This standard sorts road surfaces into classes from very good to very poor, based on how much the surface height varies. We use its classes to rate each 50 m stretch of road, so our campus map can be compared with measurements taken anywhere else.

ISO 7637-2 and SAE J1113-11 — electrical spikes in vehicles. Car electrical systems produce large voltage spikes when things switch on and off. We avoid the problem rather than solve it: we power the device from a battery pack, or through a certified 12 V-to-USB adapter that is built to handle those spikes. Our own fuse, reverse-polarity protection, and spike clamp are a second line of defense, not the first.

RoHS — restricting hazardous substances in electronics. This limits lead and several other substances in electronic assemblies. We use lead-free solder and specify RoHS-compliant parts throughout.

Illinois road law. Two statutes affect how we can test. 625 ILCS 5/12-610.2 bans handheld device use while driving, which is why the device has one button and needs no attention in motion. 625 ILCS 5/12-503 restricts objects that block the driver's view, which is why nothing gets mounted on the windshield.

3.4 Impact on Society, Money, and the Environment

Money. The expensive part of managing roads is measuring them. Setting up roadway data collection costs an average state several million dollars, so the survey vehicles come out on a multi-year cycle and only where the budget can be justified. Our parts cost under \$150. That is not just cheaper, it changes what can be measured at all. A city could put these on buses and service trucks and get continuous data on every street those vehicles touch, instead of a

snapshot of the main roads every few years. Because road damage gets worse quickly once it starts, catching it early is worth far more than the cost of measuring.

Fairness and campus impact. Survey coverage follows traffic, traffic follows investment, so the streets already being neglected are also the ones nobody measures, and the missing data becomes the excuse. Bike lanes are the clearest case: they carry the riders most at risk from bad pavement and are almost never surveyed. A device cheap enough for a student group or a neighborhood association turns complaints into evidence. Building the UIUC map shows this directly. Students would learn which routes are safe to ride, and Facilities & Services would get a ranked list instead of scattered reports.

Environment. This runs both ways and we should be honest about it. Rough pavement makes vehicles burn more fuel, and fixing roads early avoids the large amount of material and energy that goes into rebuilding one. Against that, we are making electronics, and electronics become waste. We will use lead-free assembly, keep the case openable with a removable memory card and a replaceable external battery instead of sealing everything in, and build on common parts that stay available. We will not pot the board in epoxy.

The wider world. The International Roughness Index exists because the World Bank needed a cheap, comparable measure of road quality for countries that cannot afford survey fleets. That need has not gone away. A device under \$150 that produces numbers tied to existing international standards works anywhere. We deliberately tie our scores to those standards instead of inventing our own, so our numbers can be compared with everyone else's.

References

[1] American Society of Civil Engineers, *2025 Report Card for America's Infrastructure — Roads*. Available: <https://infrastructurereportcard.org/>

[2] TRIP, *Rough Urban Roads: Motorists Pay \$130 Billion Annually in Vehicle Operating Costs*. Available: <https://tripnet.org/>

[3] AAA, *Pothole Damage Costs U.S. Drivers \$3 Billion Annually*, news release, 2016.

[4] National Highway Traffic Safety Administration, *Traffic Safety Facts: 2023 Data — Bicyclists and Other Cyclists*, National Center for Statistics and Analysis, 2025. Available: <https://crashstats.nhtsa.dot.gov/>

[5] Federal Highway Administration, *Market Analysis for Collecting Fundamental Roadway Data Elements*. Available: <https://highways.dot.gov/>

[6] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," *Scientific Reports*, vol. 3, no. 1376, 2013.

[7] T. D. Gillespie, *Fundamentals of Vehicle Dynamics*. Warrendale, PA: SAE International, 1992.

[8] *Mechanical Vibration and Shock — Evaluation of Human Exposure to Whole-Body Vibration — Part 1: General Requirements*, ISO 2631-1:1997.

- [9] *Mechanical Vibration — Road Surface Profiles — Reporting of Measured Data*, ISO 8608:2016.
- [10] *Standard Practice for Computing International Roughness Index of Roads from Longitudinal Profile Measurements*, ASTM E1926-08.
- [11] IEEE, *IEEE Code of Ethics*, IEEE Policies, Sec. 7.8. Available: <https://www.ieee.org/>
- [12] ACM, *ACM Code of Ethics and Professional Conduct*, 2018. Available: <https://www.acm.org/>
- [13] *Road Vehicles — Electrical Disturbances from Conduction and Coupling — Part 2*, ISO 7637-2:2011; and SAE J1113-11.
- [14] *Standard for Household and Commercial Batteries*, UL 2054; and *Secondary Cells and Batteries Containing Alkaline or Other Non-Acid Electrolytes — Part 2*, IEC 62133-2.
- [15] *Illinois Vehicle Code*, 625 ILCS 5/12-610.2 and 625 ILCS 5/12-503.
- [16] Bosch Sensortec, *BMI270 Datasheet*; Analog Devices, *ADXL356 Datasheet* and *ADXL1002 Datasheet*; u-blox, *MAX-M10S Data Sheet*; STMicroelectronics, *STM32F411xC/E Datasheet*; Diodes Incorporated, *AP2112 Datasheet*; Microchip, *MCP6004 Datasheet*.
- [17] OpenStreetMap contributors, *Planet OSM*. Available: <https://www.openstreetmap.org/>