



Car Accident Detection

Trasporto ideale

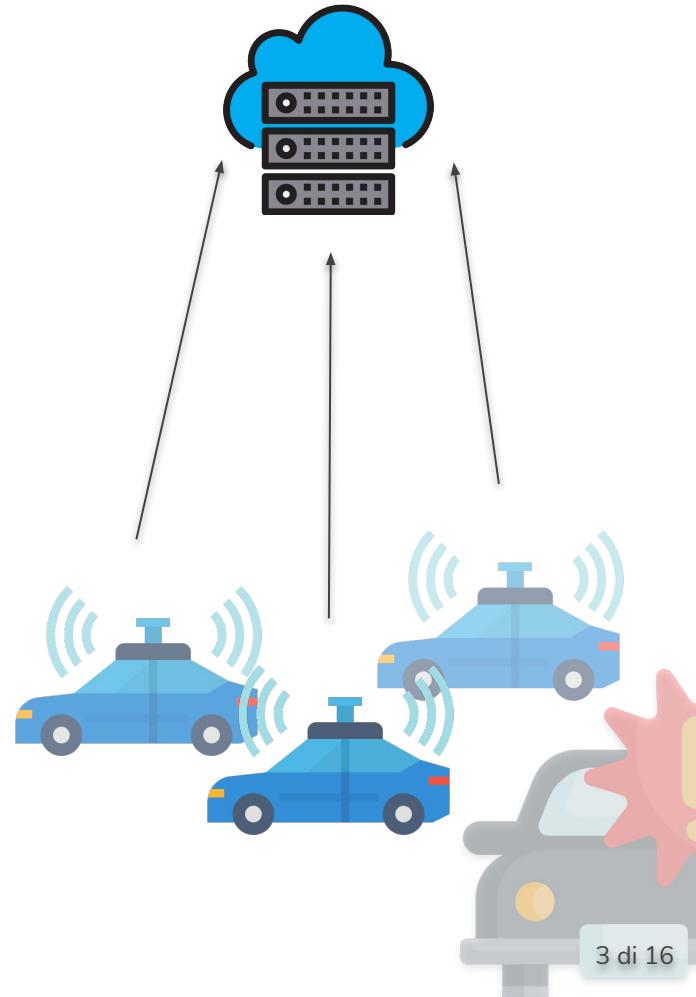
- Non subire incidenti
- Evitare lunghe code
- Puntualità

→ Dispositivo in grado di rilevare e segnalare incidenti

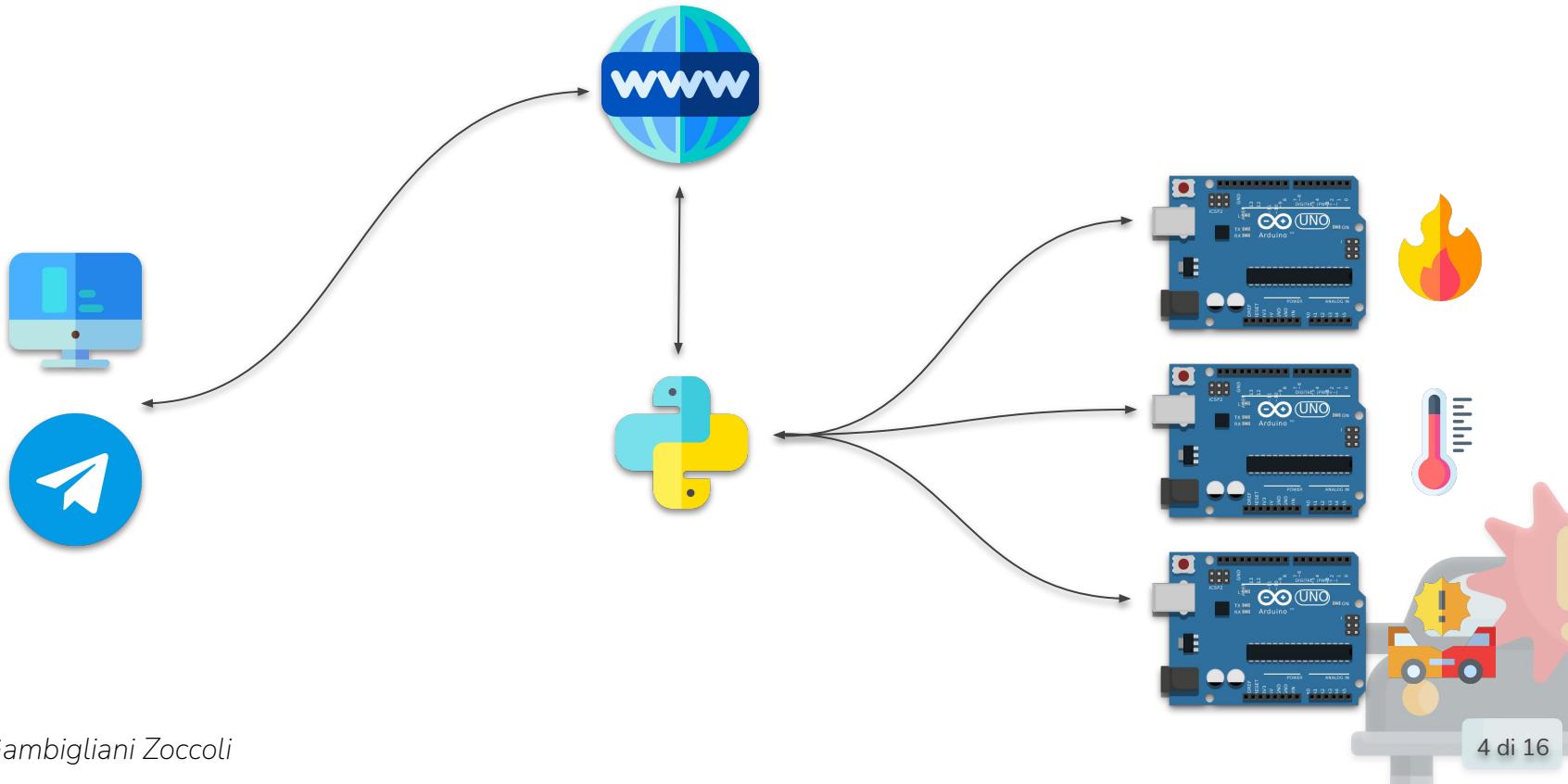
Idea

Realizzare un dispositivo in grado di:

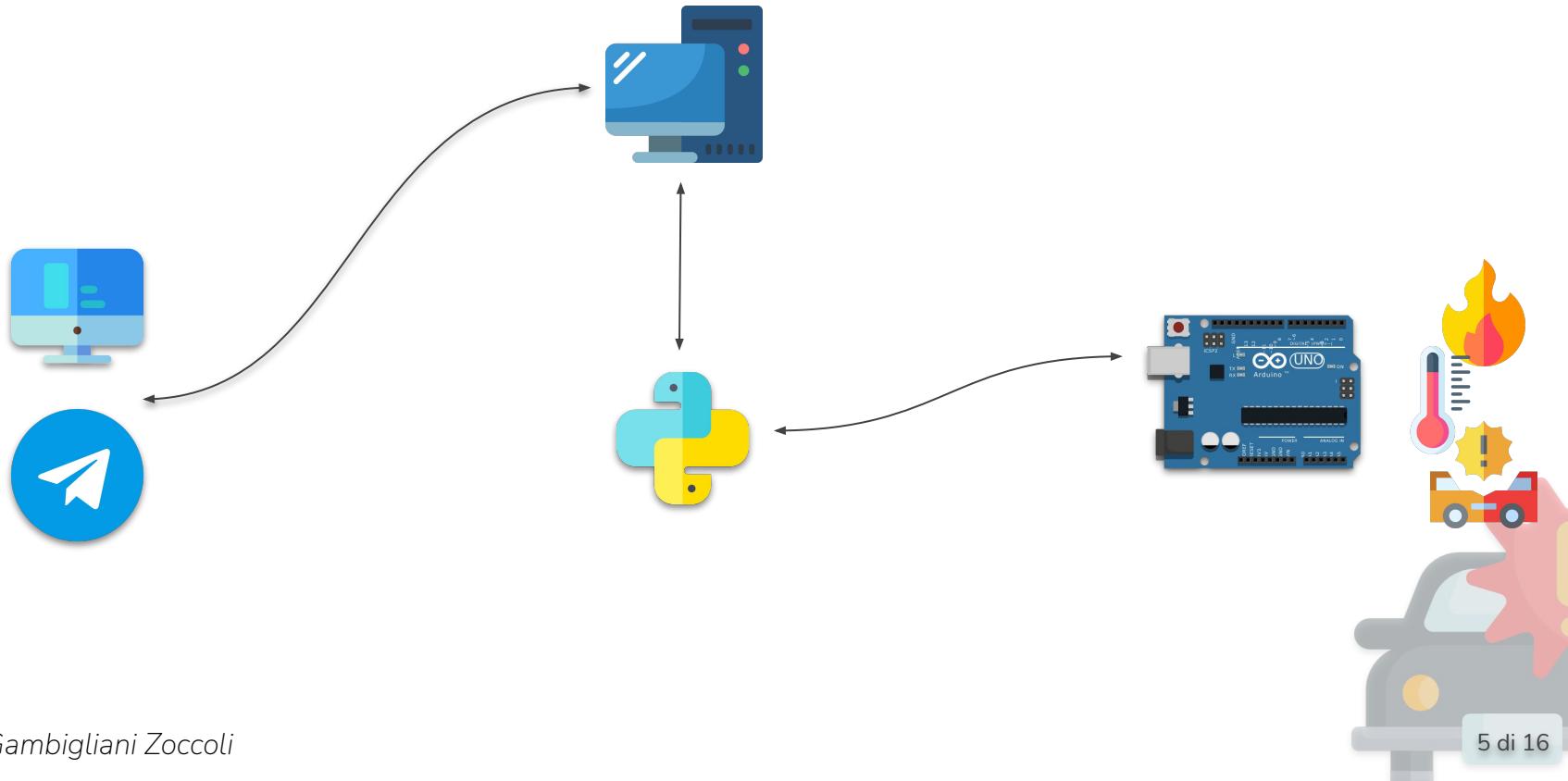
- Rilevare incidenti
- Segnalare posizione e dati sull'incidente
- Tempo reale



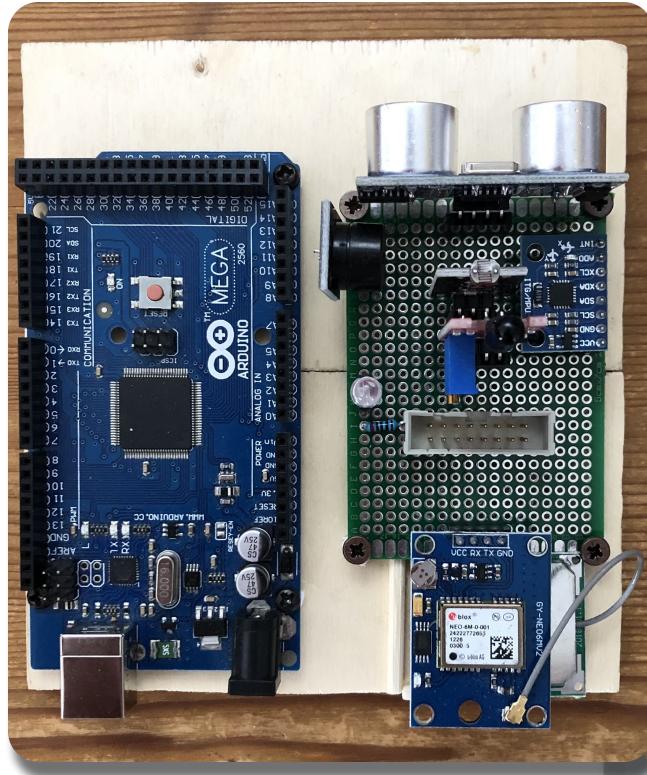
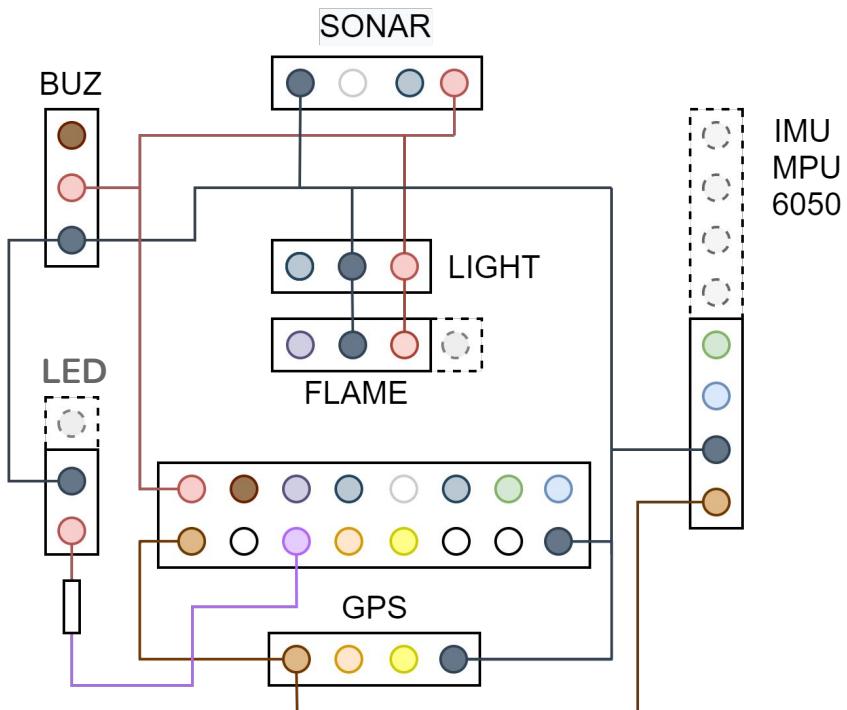
Architettura



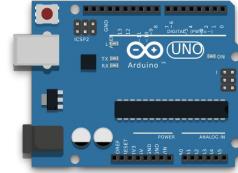
Architettura - Prototipo



Prototipo



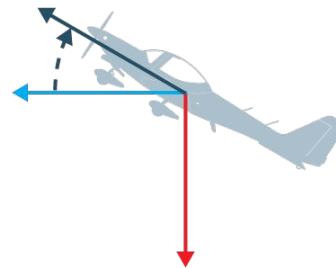
Arduino - Sensori e Attuatori



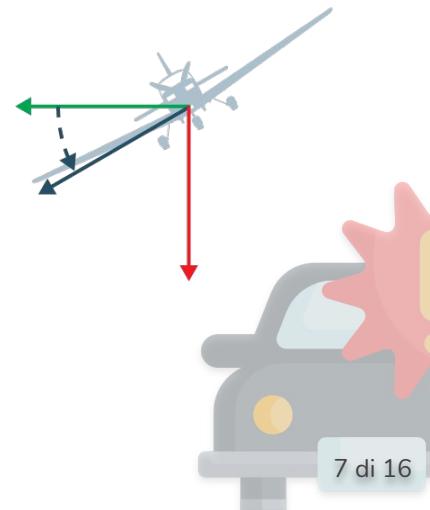
Sensori e attuatori:

- Flame → Light
- Ultrasonic (**HC-SR04**)
- Buzzer
- GPS (**NEO-6M**)
- IMU (Inertial Measurement Unit)
(GY-521)
- Led

Pitch
(around Y axis)



Roll
(around X axis)



Codici Arduino

```
bool check_parameters(){
    //Incidente frontale o tamponamento
    if(distance <= threshold.distance)  frontal = true;

    //Caduta libera
    if(AcZ + AcX + AcY < threshold.fall) fall = true;

    //Rilevazione di ribaltamento
    if(Roll >= threshold.tilt || AcZ < -threshold.fall*30) tilt = true;

    //Incidente laterale
    if(AcX - prec_acx > 1.2) detection = true;

    //Tamponamento subito
    if(AcY - prec_acy > 1.2) detection = true;

    //Controllo se è stato rilevato un fuoco
    fire = fire_read();

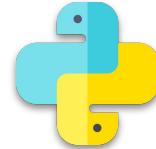
    prec_acx = AcX;
    prec_acy = AcY;
    prec_acz = AcZ;

    return (frontal | tilt | fall | fire | detection);
}
```

```
void send_flame_light_temp_data(){
    uint8_t flame = map(flame_val, 0, 1024, 0, 253);
    uint8_t light = map(light_val, 0, 1024, 0, 253);
    uint8_t temp = abs(int(Tmp));
    uint8_t data[] = {0xFE, 0x04, flame, light, temp, 0x00, 0xFF};
    if(Tmp < 0){
        data[5] = 0x01;
    }
    for (uint8_t i = 0; i < sizeof(data); i++) {
        Serial.write(data[i]);
    }
}
```

```
void accident_detected(){
    tone(buz_pin, 500);
    while(!update_gps_data()){}
    send_data();
    //Aspetto 500ms per dare il
    //tempo al bridge di rispondere
    delay(500);
    //Controllo che il bridge abbia
    //ricevuto i dati
    bridge_ack();
    noTone(buz_pin);
    //INCIDENTE SEGNALATO E RICEVUTO
    delay(60000*10); //aspetto 10 minuti
    reset_fun();
}
```

Python - Bridge

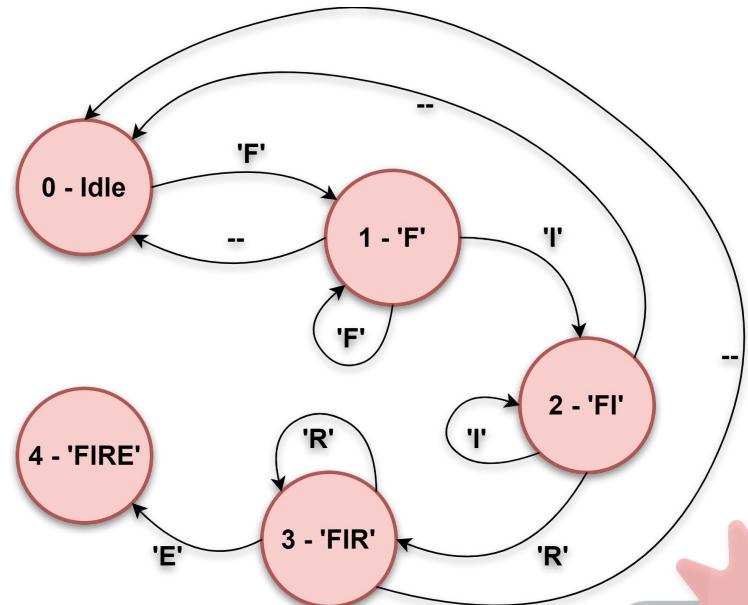


Struttura generale del pacchetto:

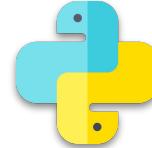


Payload:

- Flame
- Light
- Temperatura

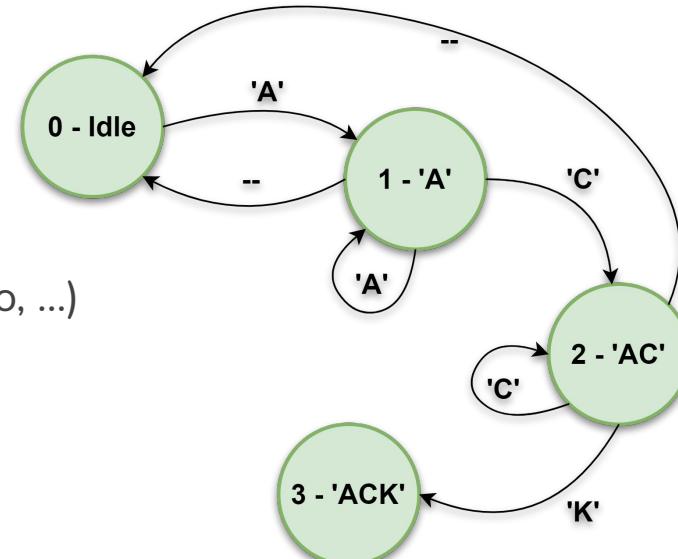


Python - Bridge



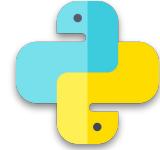
Payload:

- Posizione (Latitudine e Longitudine)
- Ora
- Data
- Informazioni sull'incidente (Frontale, Fuoco, ...)
- Temperatura
- Targa della vettura



Api Restful → **POST /api/v1/accidents**

Python - Bridge

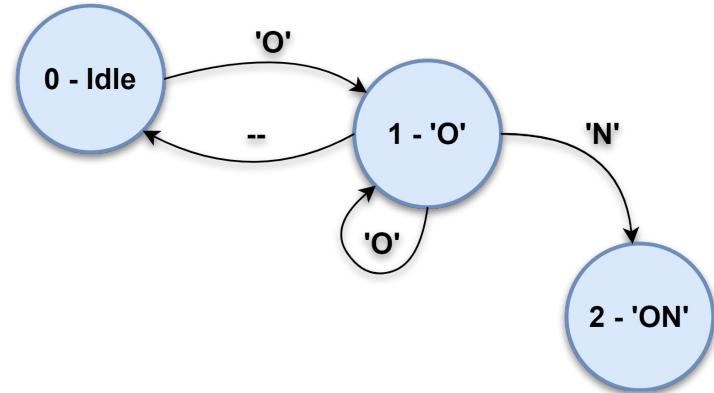


Payload:

- Posizione (Latitudine e Longitudine)
- Velocità

Raggio: 3 - 5 - 20 Km

Tempo: 10 - 20 - 30 minuti



Api Restful → **GET /api/v1/accidents**

Flask - Dashboard



```
{  
  "accidents": [  
    {  
      "date": "12/03/2021",  
      "fall": false,  
      "fire": false,  
      "frontal": true,  
      "id": 1,  
      "lat": 44.540967,  
      "license_plate": "AA 111 AA",  
      "lng": 10.931247,  
      "reported": false,  
      "temperature": 40,  
      "tilt": false,  
      "time": "19:08:08"  
    }  
  ]  
}
```

The screenshot shows a web application interface for 'Car Accident Detection'. At the top, a header bar indicates the title and a non-secure connection (<http://192.168.1.18>). The main content area is divided into two sections: 'Position' (left) and 'Car Information' (right).

Position: Displays an aerial satellite map of a rural area with fields and roads. A red exclamation mark icon is overlaid on the map at a specific location, with a tooltip showing the address: 'Str. Contrada, 45, 41126 Modena MO, Italia'. Navigation controls for the map (Google, Satellite, zoom, etc.) are visible.

Car Information: A table listing car details and accident types. The car information includes license plate (AA 111 BA), date (30/03/2021), time (09:37:43), and temperature (40 °C). The accident types listed are Fire detected (red X), Frontal Crash (green checkmark), Tilt Crash (red X), and Free Fall (red X). A button labeled 'Mark as solved' is located at the bottom right of this section.

Telegram



Azioni disponibili:

- Aggiungere una auto
- Eliminare un auto
- Visualizzare la lista della auto registrate

→ Segnalazione real-time dell'incidente

Inserisci la targa dell'auto (o /cancel per terminare): 20:11

Aa11aa 20:11✓

Auto con targa AA11AA aggiunta correttamente

20:11

Lista auto 20:11✓

Ecco la lista delle tue auto 🚗:

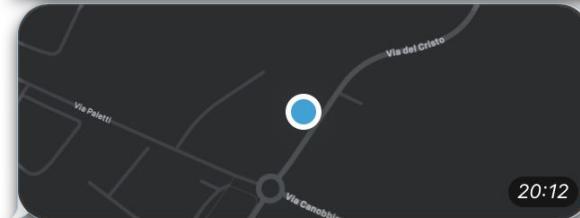
1. AA11AA

20:11

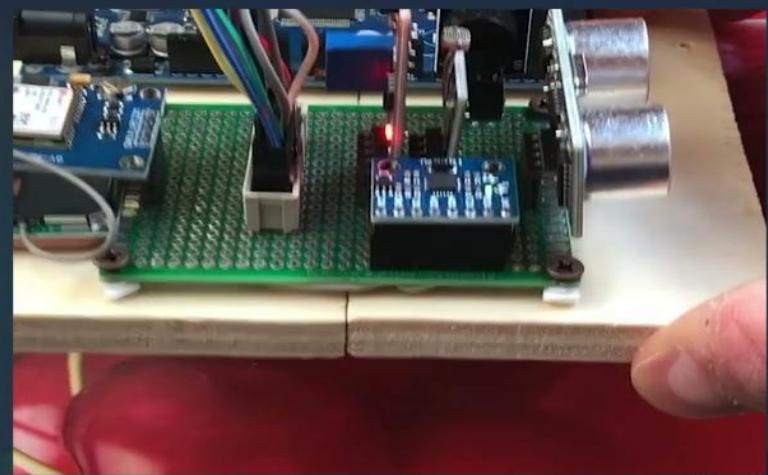
Incidente rilevato il 18/03/2021 alle 20:12:01 per l'auto con targa AA11AA.

Posizione dell'incidente:

20:12

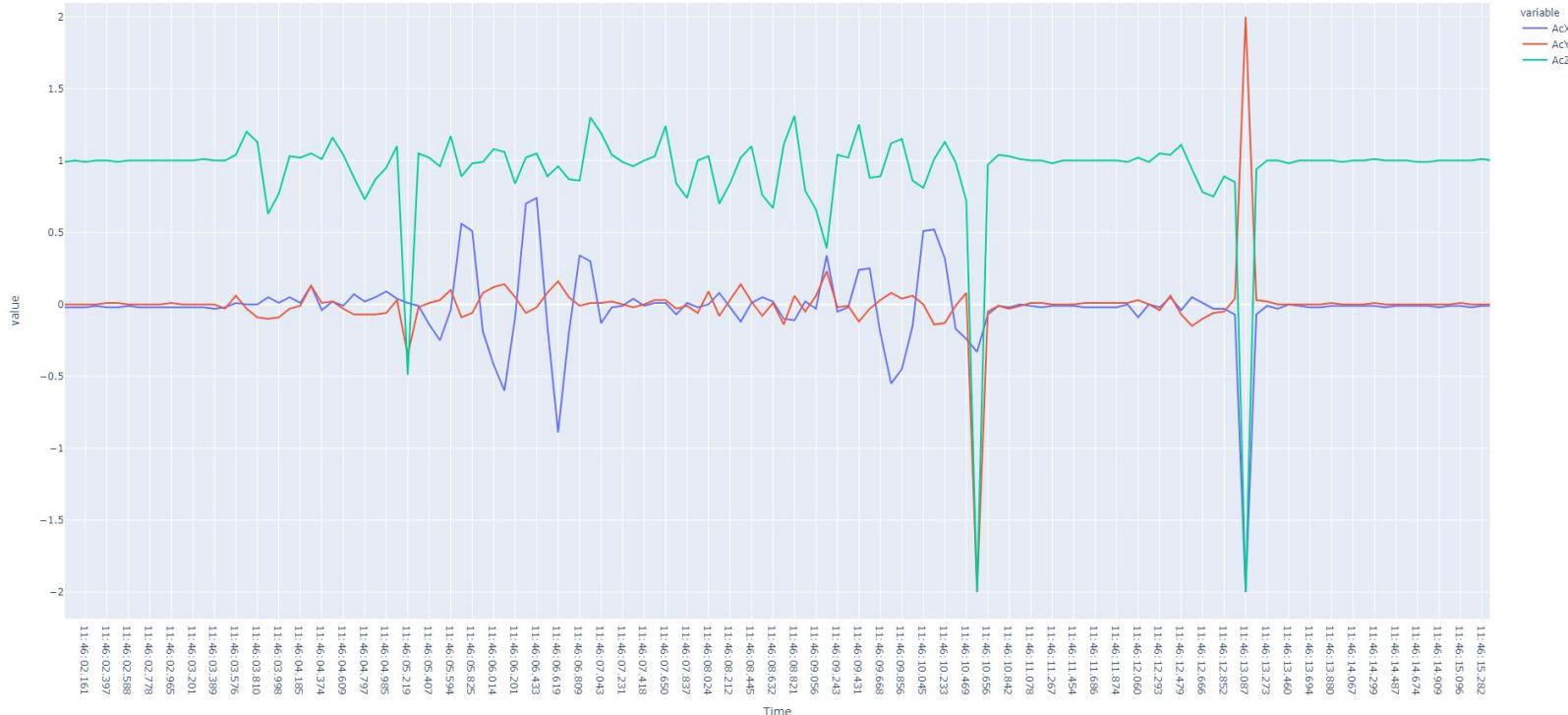


Demo...



```
(venv) E:\gitHub\car_accident_detection\python\bridge>python bridge.py
Connecting to COM3
lat=    lng=    frontal=True, tilt=False, fire=False, fall=False, tmp=21.99, targa=AA111AA, data=2021-03-23 09:19:30+01:00
lat=    lng=    frontal=False, tilt=True, fire=False, fall=False, tmp=21.94, targa=AA111AA, data=2021-03-23 09:19:34+01:00
```

Conclusioni e possibili sviluppi futuri



Grazie per l'attenzione

Giovanni Gambigliani Zoccoli