

R&D per il monitoraggio urbano – GEMA

1. Astratto

Questo progetto presenta lo sviluppo di un sistema di monitoraggio e reporting ambientale autonomo e autoriparante basato sulla piattaforma Raspberry Pi. Il sistema è progettato per rilevare rifiuti urbani e graffiti sui muri utilizzando tecniche di deep learning, mentre è montato su un veicolo che viaggia lungo percorsi predefiniti. Il Raspberry Pi cattura continuamente le immagini e le elabora localmente utilizzando una rete neurale convoluzionale basata su VGG16 per identificare la presenza di spazzatura o graffiti in tempo reale.

Ogni rilevamento viene georeferenziato utilizzando un modulo GPS (tramite SIM7000X) e i dati vengono trasmessi a un backend remoto basato su Django tramite una connessione LTE. Il sistema di back-end aggrega e visualizza le previsioni su un'interfaccia di mappa in tempo reale attraverso un front-end web, consentendo agli utenti di visualizzare tutti i rilevamenti lungo il percorso del veicolo, incluso il tipo di previsione e la cronologia delle posizioni.

Per garantire la resilienza e l'operatività del sistema in condizioni remote reali, il dispositivo include più meccanismi di watchdog. Un watchdog hardware monitora la reattività del sistema, mentre i watchdog software personalizzati rilevano sottotensione, blocchi del sistema e guasti di rete, attivando riavvii automatici quando necessario. Il sistema include anche un'API Flask per la diagnostica e le acquisizioni su richiesta.

Questo sistema mobile è ideale per le applicazioni delle smart city in cui sono richiesti il rilevamento automatizzato e la segnalazione geospaziale dei problemi di pulizia urbana, consentendo un monitoraggio e una raccolta dati efficienti senza l'intervento umano.

2. Introduzione

Gli ambienti urbani sono costantemente messi alla prova da problemi come lo scarico illegale di rifiuti e la deturpazione di spazi pubblici attraverso graffiti. Questi problemi non solo degradano il valore estetico delle città, ma pongono anche rischi per la salute e la sicurezza. Gli approcci tradizionali al monitoraggio della pulizia urbana si basano in larga misura sulle ispezioni manuali, che sono laboriose, costose e inefficienti, soprattutto in grandi aree.

Questo progetto affronta queste sfide proponendo un sistema automatizzato, mobile e basato sul deep learning per il rilevamento e la geolocalizzazione in tempo reale di rifiuti e graffiti. La soluzione è costruita su un Raspberry Pi ed è progettata per essere montata su un veicolo in movimento, come un'auto per la manutenzione urbana o un furgone di servizio. Mentre il veicolo viaggia lungo le strade urbane, il sistema di bordo acquisisce continuamente immagini, le analizza utilizzando una rete neurale convoluzionale (CNN) basata sull'architettura VGG16 e identifica le scene contenenti spazzatura o graffiti.

Oltre al riconoscimento visivo, il sistema utilizza un modulo SIM7000X per il tracciamento GPS e la comunicazione dati basata su LTE. Quando si verifica un rilevamento, il dispositivo trasmette l'etichetta prevista, un timestamp e le coordinate GPS a un'API back-end Django. Questo backend memorizza e visualizza i dati su un frontend di mappe interattivo, consentendo agli operatori di monitorare aree con alte concentrazioni di rifiuti o atti vandalici.

Per garantire un funzionamento autonomo a lungo termine in ambienti dinamici, il sistema include solide funzioni di autoripristino. Più meccanismi di watchdog, sia hardware che software, sono integrati per rilevare e risolvere problemi critici come blocchi del sistema, condizioni di sottotensione e guasti di connettività di rete. Ciò consente al dispositivo di riavviarsi automaticamente e riprendere il funzionamento senza intervento manuale, garantendo la disponibilità continua del servizio.

L'integrazione di visione artificiale, geolocalizzazione, comunicazione LTE e meccanismi di resilienza si traduce in un sistema compatto e intelligente ideale per le implementazioni nelle smart city. Fornisce una soluzione scalabile, a basso costo e in tempo reale per il monitoraggio della pulizia urbana, aprendo la strada a una manutenzione comunale basata sui dati e a tempi di risposta più rapidi.

3. Architettura del sistema

L'architettura di sistema della piattaforma di rilevamento e reporting intelligente basata sul deep learning è progettata per essere modulare, robusta e adatta all'implementazione mobile. Si compone di cinque sottosistemi principali che lavorano in parallelo:

3.1. Panoramica

Al centro del sistema c'è un computer a scheda singola Raspberry Pi, che orchestra l'acquisizione delle immagini, l'inferenza del deep learning, l'acquisizione dei dati GPS e la comunicazione con il server di back-end. Il sistema viene implementato su un veicolo in movimento e funziona in modo autonomo, elaborando dati in tempo reale e riportando i risultati senza l'intervento umano.

L'architettura include i componenti seguenti:

- Sottosistema di acquisizione immagini
- Motore di inferenza di deep learning
- Sottosistema Geolocalizzazione e Comunicazione
- Sottosistema watchdog e ripristino automatico
- Livello di integrazione back-end

3.2. Sottosistema di acquisizione delle immagini

- Utilizza il modulo fotocamera Raspberry Pi controllato tramite `libcamera-still`.
- Acquisisce i fotogrammi a intervalli regolari o in modo continuo.
- Memorizza temporaneamente le immagini acquisite in una coda thread-safe per l'elaborazione.
- Supporta il doppio accesso (continuo e su richiesta) utilizzando meccanismi di blocco dei file per evitare conflitti tra gli script.

3.3 Motore di inferenza di apprendimento profondo

- Costruito con TensorFlow e una rete neurale convoluzionale basata su VGG16, pre-addestrato e ottimizzato per classificare:
 - Garbage (rifiuti)

- Graffito
 - Pulito / Nessun rilevamento
- Esegue la pre-elaborazione dei frame, la previsione e il punteggio di attendibilità.
- Se una classe pertinente viene rilevata con sufficiente sicurezza, il risultato viene inviato alla pipeline di trasmissione.

3.4 Sottosistema Geolocalizzazione e Comunicazione

- Utilizza SIM7000X modulo per:
 - GPS: Acquisisce le coordinate di latitudine e longitudine.
 - LTE: invia i dati di previsione e posizione al back-end.
- La trasmissione dei dati avviene tramite API RESTful a un backend basato su Django, che memorizza le informazioni in un database PostgreSQL e le serve a una dashboard di mappe frontend.
- Trasmette periodicamente anche l'indirizzo IP pubblico del dispositivo e le coordinate GPS correnti per il tracciamento in tempo reale.

3.5 Sottosistema watchdog e ripristino automatico

Per garantire un funzionamento continuo in condizioni di campo:

- Hardware Watchdog (/dev/watchdog): Riavvia automaticamente il Pi se non risponde.
- Cani da guardia del software:
 - Un file heartbeat personalizzato aggiornato dallo script Python principale.
 - Uno script di rilevamento della sottotensione basato su crontab che riavvia il dispositivo se vengono rilevati codici di limitazione critici (0x1, 0x50000 e così via) tramite `vcgencmd get_throttled`.
 - Un watchdog di rete (facoltativo) che verifica la perdita di connettività mobile e attiva un riavvio se la connessione non riesce in modo persistente.

3.6 Integrazione backend

- Il server di backend è costruito con Django REST Framework e include:
 - Endpoint API per la ricezione di previsioni, posizione e dati IP.

- Un'interfaccia front-end che visualizza i rilevamenti su una mappa con una linea di percorso per ogni dispositivo.
- Il server contrassegna e marca temporale ogni invio, consentendo il monitoraggio dei dati cronologici e dell'integrità del dispositivo:
 - Stato: Online/Offline.
 - Modalità: inattiva/in movimento.
 - Ultimo attivo che mostra il timestamp dell'ultima posizione ricevuta.

3.7 Design filettato per il funzionamento simultaneo

Il sistema Raspberry Pi genera diversi thread in parallelo per gestire:

- Acquisizione continua di fotogrammi
- Elaborazione e classificazione dei frame
- Lettura dati GPS
- Invio di stime e dati di telemetria
- Calci Watchdog (hardware)
- Scrittura di file heartbeat (software)

Questa architettura garantisce prestazioni in tempo reale, isolamento dei guasti tra i componenti e un uso efficiente delle risorse hardware limitate.

4. Progettazione e implementazione del backend

Il sistema di back-end è responsabile della ricezione, dell'archiviazione e dell'elaborazione dei dati inviati dai dispositivi Raspberry Pi. Viene implementato utilizzando il framework REST Django per creare un'API robusta che gestisce gli eventi di rilevamento in entrata, tra cui immagini, coordinate GPS e timestamp.

I dati vengono archiviati in un database relazionale che conserva le informazioni sul dispositivo, i record di rilevamento e i metadati associati. Il back-end fornisce anche endpoint per l'interrogazione della cronologia dei rilevamenti e dello stato corrente del dispositivo.

Sono in atto meccanismi di autenticazione e sicurezza per garantire che solo i dispositivi e gli utenti autorizzati possano inviare o accedere ai dati.

5. Visualizzazione frontend e interfaccia utente

Il frontend è progettato per fornire una visualizzazione intuitiva e interattiva dei dati raccolti per gli operatori e le parti interessate.

Le caratteristiche principali includono:

- Un' interfaccia mappa che traccia le posizioni dei rifiuti e dei graffiti rilevati in tempo reale, con indicatori cliccabili visualizzati.
- Visualizzazione del percorso percorso da ciascun dispositivo Raspberry Pi, che consente agli utenti di tracciare i percorsi durante la raccolta dei dati utilizzando WebSocket.
- Filtri e opzioni di ricerca per sfogliare i rilevamenti per data, dispositivo o tipo con la possibilità di scaricare i dati come file CSV.
- Scheda dispositivi per controllare lo stato, la modalità e l'ultima attività del raspberry pi in tempo reale utilizzando WebSocket, inoltre la possibilità di testare la fotocamera sul dispositivo e mostrarne la posizione.

Il front-end recupera i dati dalle API di back-end e li aggiorna in modo dinamico, garantendo agli utenti informazioni aggiornate sugli sforzi di monitoraggio ambientale.

6. Distribuzione e test del sistema

L'implementazione prevede l'installazione e la configurazione dei dispositivi Raspberry Pi nei veicoli, garantendo un'alimentazione adeguata, l'allineamento della telecamera e la connettività di rete.

Lo stack software è containerizzato o confezionato per facilitare gli aggiornamenti e la manutenzione. I servizi di Systemd gestiscono i processi critici per abilitare il riavvio automatico in caso di errore.

I test includono:

- Unit test per gli endpoint API back-end.

- Test di integrazione che verificano il flusso di dati end-to-end dal rilevamento alla visualizzazione.
- Test sul campo con viaggi su strada reali per convalidare l'accuratezza del rilevamento, la resilienza della connettività e la stabilità del dispositivo.
- Stress test per garantire che il sistema gestisca più dispositivi che trasmettono contemporaneamente senza perdita di dati.

Il monitoraggio e la registrazione continui aiutano a identificare problemi come eventi di sottotensione, interruzioni di rete o guasti della telecamera, consentendo una manutenzione e miglioramenti proattivi.

7. Risultati e analisi

Il sistema ha rilevato con successo rifiuti e graffiti durante i viaggi di prova, catturando con precisione le immagini con le coordinate GPS. Il back-end ha elaborato e archiviato gli eventi di rilevamento in modo affidabile, mentre le visualizzazioni delle mappe front-end hanno fornito rappresentazioni chiare e di facile utilizzo dei dati.

Le metriche delle prestazioni mostrano che il modello ha mantenuto l'elaborazione in tempo reale a circa 30 fotogrammi al secondo, consentendo un monitoraggio continuo senza ritardi significativi. La connettività di rete è rimasta generalmente stabile, anche se sono stati osservati cali occasionali nelle aree con segnale mobile debole.

Le condizioni di sottotensione sono state rilevate dagli script watchdog, che hanno attivato con successo il riavvio del sistema per mantenere il tempo di attività.

Nel complesso, il sistema ha dimostrato solide capacità di rilevamento e una comunicazione affidabile, soddisfacendo gli obiettivi del progetto.

8. Sfide e soluzioni

- **Conflitti di risorse hardware:** l'accesso alla telecamera da più thread richiedeva l'implementazione di meccanismi di blocco basati su file per evitare conflitti tra l'acquisizione continua e le richieste di snapshot su richiesta.

- **Stabilità dell'alimentazione:** la sottotensione causava blocchi imprevisti; risolta integrando watchdog hardware e script di rilevamento della sottotensione personalizzati che forzano il riavvio automatico.
- **Connettività di rete:** le interruzioni dell'hotspot mobile sono state gestite con uno script watchdog di rete basato su ping che attiva il riavvio in caso di perdita di connessione prolungata.
- **Acquisizione del rilevamento GPS:** la garanzia che il dispositivo ottenga un rilevamento GPS stabile prima di iniziare la raccolta dei dati è stata ottenuta tramite loop di blocco e tentativi.
- **Monitoraggio del sistema:** sono stati implementati registri e file heartbeat watchdog per consentire il monitoraggio e il debug remoti.

9. Conclusione

Questo progetto dimostra un approccio pratico e scalabile al monitoraggio ambientale automatizzato utilizzando dispositivi Raspberry Pi dotati di telecamere e GPS.

Combinando il rilevamento basato sul deep learning con solide soluzioni di gestione del sistema e connettività, il sistema fornisce dati tempestivi e accurati su rifiuti e graffiti lungo la strada.

L'integrazione di meccanismi di watchdog garantisce un'elevata disponibilità anche in condizioni di campo difficili. I miglioramenti futuri potrebbero includere modelli di rilevamento migliorati, suite di sensori ampliate e analisi front-end più sofisticate.

Questo lavoro getta solide basi per applicazioni di smart city volte a migliorare la pulizia urbana e l'efficienza della manutenzione.

Mappa

Dispositivi

Cronologia

Profilo

Esci

powered by SOMO

Mappa delle Previsioni dei Dispositivi

Dispositivo: device-001 Previsione: Tutte Mostra

Legenda

- Percorso
- Graffiti
- Rifiuti

Mappa

Dispositivi

Cronologia


Profilo

Esci

powered by SOMO

Stato Del Dispositivo

ID	STATO	MODALITÀ	ULTIMA ATTIVITÀ	IMPOSTAZIONI
device-001	Online	Inattivo	2025-07-03 10:59:47	




Mappa

Dispositivi


Cronologia

Profilo


Esci


powered by 

Stato Del Dispositivo

ID	STATO	MODALITÀ	ULTIMA ATTIVITÀ	IMPOSTAZIONI
device-001	Online	Inattivo	2025-07-03 10:59:47	

Test Fotocamera






Mappa

Dispositivi

Cronologia

Profilo

Esci

powered by 

Stato Del Dispositivo

ID	STATO	MODALITÀ	ULTIMA ATTIVITÀ	IMPOSTAZIONI
device-001	Online	Inattivo	2025-07-03 10:59:47	

Posizione del Dispositivo


+

-



Dispositivo: device-001

 Leaflet | © OpenStreetMap contributors




Mappa

Dispositivi

Cronologia

Profilo

Esci

powered by 


Storico delle Previsioni

Previsione Latitudine Longitudine Da A

Scarica CSV

Dispositivo	Previsione	Latitudine	Longitudine	Data e Ora
device-001	Graffiti	39.368307	16.225001	7/3/2025, 10:59:50 AM
device-001	Rifiuti	39.334654	16.241436	7/2/2025, 5:04:50 PM
device-001	Graffiti	39.334654	16.241436	7/2/2025, 5:04:48 PM
device-001	Rifiuti	39.334654	16.241436	7/2/2025, 5:03:42 PM
device-001	Graffiti	39.334654	16.241436	7/2/2025, 5:03:36 PM
device-001	Rifiuti	39.334654	16.241436	7/2/2025, 5:03:23 PM
device-001	Graffiti	39.334654	16.241436	7/2/2025, 5:03:20 PM
device-001	Rifiuti	39.334654	16.241436	7/2/2025, 5:02:15 PM
device-001	Graffiti	39.334654	16.241436	7/2/2025, 5:02:14 PM
device-001	Rifiuti	39.334654	16.241436	7/2/2025, 5:01:55 PM

Pagina 1 di 51




Mappa

Dispositivi

Cronologia

Profilo

Esci

powered by 

Profilo

Nome utente

Email

Telefono

Password

Disattiva account

Elimina account