Relazione di Progetto

Scansione 3D e Analisi di Pezzi Meccanici: Sviluppo di una Pipeline di Computer Vision

Introduzione e Obiettivi del Progetto

Il presente documento descrive le attività di ricerca e sviluppo condotte dal gruppo Polimi Data Scientists nell'ambito di una collaborazione con Devnut S.R.L.. Il progetto si è concentrato sulla creazione di una pipeline software in grado di digitalizzare e analizzare oggetti fisici tramite la loro scansione, al fine di supportare il controllo qualità e la gestione della produzione di pezzi meccanici.

L'obiettivo principale è stato lo sviluppo di una soluzione software che permetta di elaborare i dati acquisiti per generare modelli 3D e confrontarli con i file di progetto originali (in formato `.STEP`), verificandone la conformità in termini di accuratezza, precisione e tolleranze di fabbricazione. Questo approccio mira a identificare tempestivamente le criticità nel processo di produzione.

L'applicazione finale è stata concepita per consentire agli utenti di un'azienda, previa abilitazione da parte di un utente con privilegi, di scansionare oggetti fisici utilizzando la fotocamera del telefono. La soluzione completa si compone di una mobile app e una web app, che interagiscono con un backend per l'elaborazione dei dati.

Il server backend si disloca su Firebase per quanto riguarda la gestione delle funzioni prettamente applicative e di coordinazione con database e Cloud Storage, mentre la logica di pura analisi dell'elaborato 3D avviene su un server sicuro locale, sempre in contatto con l'applicativo Firebase.

Dettagli del Team e Organizzazione

Il team di sviluppo è composto da Alessandro, Vittorio, Beatrice, Jacopo e Stefano, tutti studenti del Politecnico di Milano e membri del gruppo Polimi Data Scientists. Il ruolo di Project Manager è stato gestito a rotazione: inizialmente ricoperto da Jacopo, è passato a Beatrice e infine a Stefano, a seguito della riorganizzazione del team. Tutti i membri hanno contribuito attivamente come developer, dividendosi mansioni e fasi di elaborazione per ottimizzare al meglio le risorse.

Fasi di Sviluppo

Il progetto è stato suddiviso in diverse fasi principali:

Fase di Ricerca e Acquisizione Dati (Gennaio-Febbraio)

La fase iniziale è stata dedicata alla raccolta dei dati e all'esplorazione delle tecnologie più appropriate. Sono stati realizzati prototipi di pezzi con una stampante 3D e sono stati acquisiti i primi video. Parallelamente, è stata condotta una ricerca approfondita su librerie e modelli di computer vision, con l'obiettivo di valutare e confrontare diverse soluzioni.

Tra le tecnologie prese in considerazione rientravano SfM + MVS e NeRF, metodologie promettenti in letteratura ma di difficile applicazione pratica in contesti industriali: i tempi di elaborazione si sono rivelati proibitivi e la qualità dei risultati non garantiva tolleranze sufficienti. Queste sperimentazioni hanno comunque fornito insight importanti sui limiti delle tecniche basate su pura ricostruzione da immagini.

Sviluppo e Risoluzione di Problematiche Tecniche

Durante lo sviluppo sono emerse diverse sfide tecniche. Il problema principale è stato l'elevato carico computazionale richiesto dai modelli di computer vision: sono state adottate soluzioni per ottimizzare le risorse, come la connessione a un ambiente cloud dedicato o l'uso di GPU remote. Per migliorare i tempi di elaborazione, sono state implementate tecniche di preprocessing delle immagini, tra cui la riduzione del numero di frame, la diminuzione delle dimensioni delle immagini e l'applicazione di filtri in bianco e nero.

Il team ha inoltre implementato un sistema per lo sviluppo del codice tramite l'estensione CLINE su Visual Studio, configurata con un LLM come Gemini, per supportare l'automazione delle fasi di testing e refactoring.

Architettura Software e Sviluppo della Piattaforma

L'architettura del software è stata delineata per supportare un flusso utente completo, dalla registrazione alla visualizzazione dei risultati. Le tecnologie scelte sono state:

- **Frontend**: Flutter
- **Backend**: Python/Flask
- **Database**: Firebase (Firestore per i dati, Cloud Storage per i file, Authentication per il login)

Il modello di database su Firestore è stato progettato per gestire in tempo reale gli stati delle scansioni (`scans`), i dati statistici delle analisi (`stats`), i file di riferimento (`steps`) e i profili degli utenti (`users`). Le regole di sicurezza e i trigger automatici (tramite Cloud Functions) sono stati implementati per gestire l'intero ciclo di vita dei dati.

L'applicazione si articola in due componenti principali:

- **Web App**: Gestisce le funzioni amministrative, permettendo agli utenti con livello 1 di caricare e gestire i file `.STEP` e a quelli con livello 2 di gestire gli utenti del sistema.
- **Mobile App**: Consente agli utenti di scansionare i pezzi e di selezionare il file `.STEP` di riferimento. I dati della scansione vengono inviati al server, che avvia l'elaborazione. Una volta completata, l'app mobile visualizza i risultati, inclusa una heatmap interattiva che mostra in dettaglio le deviazioni e le accuratezze del pezzo scansionato rispetto al modello CAD. Il visualizzatore integra controlli di rotazione e zoom, per un'esplorazione dinamica, ma non offre ancora confronto tra scansioni multiple né esportazioni.

Transizione Tecnologica: Da Flusso Video a LiDAR

Inizialmente, il progetto prevedeva l'utilizzo di un flusso video standard per la scansione degli oggetti. Tuttavia, dopo mesi di ricerca e sviluppo, il team ha riscontrato che i risultati ottenuti da questa metodologia erano imprecisi e non idonei a garantire l'accuratezza richiesta per il controllo qualità. Le problematiche principali includevano sensibilità all'illuminazione, limiti di performance su dispositivi Android e difficoltà nell'allineamento delle immagini.

Per superare queste limitazioni e garantire la massima affidabilità, è stato deciso di virare verso l'utilizzo del sensore LiDAR presente sugli iPhone. Il LiDAR permette un'acquisizione di dati 3D molto più rapida e precisa, generando nuvole di punti di alta qualità che sono ideali per le successive fasi di analisi. Questo passaggio ha rappresentato un turning point cruciale, consentendo di ridurre drasticamente i tempi medi di acquisizione e aumentando la coerenza tra scansioni ripetute.

Tecnologie Utilizzate

Hardware e Infrastruttura di Sviluppo

- **Mobile App**: sviluppo e testing condotti principalmente su iPhone Pro dotati di sensore LiDAR, sfruttando la capacità di acquisizione dati 3D ad alta precisione.
- **Backend**: ospitato su un server locale presso la sede di Devnut, utilizzando Docker per garantire un ambiente di esecuzione isolato e portabile.
- **Cloud**: database e servizi di autenticazione gestiti tramite Firebase; i file delle scansioni e i file di riferimento `.STEP` archiviati su Firebase Cloud Storage.

Flusso di Lavoro della Pipeline

La comunicazione tra i vari componenti dell'architettura è gestita in maniera asincrona, con Firebase che funge da intermediario. L'app mobile carica la scansione (ad esempio un file `.ply`) su Firebase Cloud Storage. A seguito di questo caricamento, una Firebase Cloud Function (`process_user_scan`) viene attivata automaticamente.

Questa funzione ha il compito di registrare la scansione nel database Firestore e generare degli URL temporanei e firmati per i file `.ply` (della scansione) e `.step` (di riferimento). Tali URL vengono poi inviati al backend hostato su Docker, che può così scaricare in modo sicuro i file e avviare l'elaborazione. Questo approccio garantisce che i dati rimangano protetti e che l'accesso avvenga solo tramite canali autorizzati.

Una volta che il backend ha terminato l'elaborazione dei dati e la generazione del file delle statistiche, questo viene caricato su Firebase Cloud Storage. A questo punto, un'altra Firebase Cloud Function (new_stats) viene attivata per inviare una notifica push all'utente, informandolo che l'elaborazione è stata completata.

- Tempi di elaborazione eccessivi per un contesto industriale.
- Sensibilità alle condizioni di luce e qualità video.
- Scarsa riproducibilità delle tolleranze richieste.

Questi esperimenti, pur fallimentari dal punto di vista pratico, hanno chiarito i requisiti minimi necessari per garantire risultati affidabili e hanno motivato la transizione verso il LiDAR.

Metriche e Valutazioni Empiriche

Non sono state formalizzate metriche standard, ma il team ha monitorato parametri empirici utili a valutare i progressi:

- **Tempo medio di scansione**: ridotto da diversi minuti (metodo video) a meno di un minuto (LiDAR).
- **Grado di accuratezza**: valutato visivamente tramite heatmap e comparazione con CAD, con deviazioni mediamente inferiori al millimetro sui prototipi stampati in 3D.
- **Affidabilità del processo**: numero di scansioni valide vs fallite, con un incremento netto della percentuale di successi dopo la transizione al LiDAR.

Esempi di Codice: Gestione dei Flussi

Trigger di Elaborazione Scansione

La funzione `process_user_scan` è un esempio chiave di come la pipeline viene attivata. Si attiva ogni volta che un nuovo file di scansione viene caricato su Cloud Storage e ha il ruolo di avviare il processo di analisi nel backend. L'implementazione gestisce l'ottenimento di URL firmati per garantire un accesso sicuro ai file. Il file `.STEP` di riferimento da utilizzare viene specificato nei metadati della scansione e recuperato da Firestore.

Invio di Notifiche (`new_stats`)

Questa funzione Firebase Cloud Function si attiva quando un documento viene creato nella collezione stats. Il suo ruolo è inviare una notifica push all'utente che ha effettuato la scansione. La funzione recupera l'user_id dal documento delle statistiche e usa questo ID per cercare il token FCM dell'utente nella collezione users.

Ruoli e Funzionalità Utente

L'applicazione gestisce tre diversi livelli di privilegio utente, ognuno con funzionalità specifiche:

- Livello 0 (Utente Base): L'utente apre l'app mobile, si autentica (se necessario), e visualizza una lista delle scansioni precedenti. Un pulsante "+" gli permette di avviare una nuova scansione. L'app richiede all'utente di dare un nome alla scansione e di selezionare il file `.STEP` di riferimento. Al termine della fase di acquisizione LiDAR, l'app presenta un riepilogo prima di procedere con l'invio al server. L'app mobile mostra i risultati delle scansioni passate con una visualizzazione 3D interattiva che permette di ruotare e zoomare l'oggetto.

- Livello 1 (Amministratore degli "Step"): Oltre a tutte le funzionalità del Livello 0, questi utenti hanno accesso esteso tramite la web app. Possono visualizzare le scansioni di tutti gli utenti, e hanno la possibilità di caricare nuovi file `.STEP` o di eliminarli. Dalla web app è anche possibile scaricare il file .PLY elaborato.

- Livello 2 (Superuser): Questo è il livello di privilegio più alto. Oltre a tutte le funzionalità dei livelli 0 e 1, un superuser può aggiungere nuovi utenti, abilitarli o disabilitarli tramite l'interfaccia della web app. Le azioni più rischiose, come la "pulizia" del database per rimuovere tutte le scansioni e le statistiche, sono protette da un sistema di prompt che richiede una doppia conferma visiva per evitare cancellazioni accidentali.

Pipeline di Analisi: Funzionamento e Tecnologie

La pipeline di analisi è il cuore del sistema, responsabile della trasformazione dei dati grezzi acquisiti dal LiDAR in un report di analisi dettagliato. Sebbene la fase di test sia ancora in corso, la struttura è stata definita e le librerie chiave sono state selezionate. La pipeline si articola nelle seguenti fasi:

- Gestione degli Errori: Per garantire la robustezza del sistema, la pipeline include meccanismi di gestione degli errori. Ad esempio, se l'elaborazione di una scansione fallisce (a causa di un file corrotto o di un allineamento non riuscito), il backend aggiorna lo stato della scansione su Firestore, portandolo a "errore". Questo permette all'app mobile di visualizzare un messaggio all'utente per informarlo del problema, anziché rimanere bloccata in attesa.

- Flusso dei Dati: La pipeline è composta da diversi script Python (extracted_component.py, alignment.py, ecc.) che vengono eseguiti in sequenza. L'output di ogni script (ad es. una nuvola di punti pulita da extracted_component.py) diventa l'input per lo script successivo. L'orchestrazione di questo flusso è

gestita dal server di backend, che riceve i file e i metadati da Firebase e lancia gli script in sequenza, passando i file temporanei da una fase all'altra.

- Ottimizzazione dei Parametri: La messa a punto dei parametri di algoritmi come RANSAC, DBSCAN e ICP è cruciale per la precisione. Essi sono stati ottimizzati empiricamente. A causa della transizione al sensore LiDAR, questa fase di calibrazione è in attesa del completamento della parte di scansione, ma sono previsti test dedicati per trovare i valori ottimali per diversi tipi di oggetti.

Gestione del Carico e Parallelismo

Per gestire il carico di lavoro in parallelo, il server backend è stato progettato per utilizzare un sistema di thread. Quando un endpoint riceve una richiesta di elaborazione, delega il compito a un worker che opera in un thread separato. Questo approccio permette al server di accettare nuove richieste senza aspettare che l'elaborazione precedente sia completata, ottimizzando così l'utilizzo delle risorse.

Sicurezza e Accesso ai Dati

La sicurezza dei dati è garantita a vari livelli. I file sensibili su Cloud Storage sono protetti da un sistema di autenticazione e accesso tramite URL firmati, che scadono dopo un breve periodo. Il server di backend stesso accede a questi URL tramite un API key, garantendo che solo un'entità autorizzata possa accedere e processare i dati. Inoltre, il server è configurato per utilizzare un token di autenticazione per le chiamate API in ingresso, come mostrato nel codice di routes.py, proteggendolo da accessi non autorizzati.

Prossimi Passi

Il progetto è in continua evoluzione, e i prossimi passi sono chiaramente definiti:

- Validazione e Calibrazione della Pipeline: Dopo la transizione al LiDAR, la priorità è calibrare e testare a fondo la pipeline di analisi. Questo include la messa a punto dei parametri degli algoritmi e la creazione di test dedicati per garantire che l'intero flusso funzioni in modo accurato e stabile.

- Perfezionamento del Backend: L'implementazione del parallelismo tramite thread è un primo passo, ma il team prevede di valutare l'adozione di un sistema di coda di messaggi per una gestione del carico ancora più robusta e scalabile.

- Miglioramento dell'Analisi: Oltre alla semplice generazione di statistiche, sono in programma l'implementazione di algoritmi per la pulizia delle immagini, l'elaborazione avanzata dei modelli e il "gap filling" per colmare le aree mancanti nelle scansioni. Verranno inoltre introdotti test di accuratezza più specifici per una valutazione più completa.