
# Tesi di Laurea

Pietro Bernabei - matricola:6291312

Anno Accademico 2019/20

# 1 Introduzione

## 1.1 Motivazioni

L'avanzamento tecnologico vissuto nell'ultimo periodo, ha portato ha un cambiamento nelle nostre vite, con l'introduzione di nuovi sistemi, applicazioni e infrastrutture, hardware e software, sulle quali si fa sempre più affidamento. Data questa particolare dipendenza dalla loro presenza, si richiede che questi siano continuativi, liberi da possibili errori o malfunzionamenti. Questi sono definiti Sistemi critici. Un particolare sistemi critico, che verrà trattato da questa tesi, è il sistema di guida autonoma di una macchina. Tecnologia in sviluppo negli ultimi anni prevede l'uso di sensori, per interagire e sentire il mondo esterno. Uno dei più importanti e comuni sensori usati all'interno dei sistemi a guida autonoma, sono le telecamere." Dispositivo elettronico in grado di acquisire immagini bidimensionali in sequenza a velocità di cattura prefissate, solitamente nella gamma visibile dello spettro elettromagnetico"["wikipedia-telecamere"]. Questo può subire diversi malfunzionamenti, come la sfuocatura, il congelamente del vetro, la rottura dei pixels, e altri ancora, i quali portano a malfunzionamenti nel momento decisionale.

## 1.2 Obbiettivo

Detto ciò, l'obbiettivo della seguente tesi è di andare a definire un sistema software, detto detector, in grado di andare a rilevare, nel flusso di immagini rilevate in input dalla telecamara del sistema, i malfunzionamenti nel sistema di acquisizione.

## 1.3 Organizzazione del lavoro

Il detector sviluppato nella seguente tesi, si basa sull'implementazione di una intelligenza artificiale, CNN (convulutional Neureal Network), addestrata con un dataset di immagini acquisite in precedenza, per poi essere inserito all'interno del simulatore

# 2 Fondamenti

## 2.1 Introduzione

Il rapido avanzamento tecnologico a cui assistiamo ogni giorno ha introdotto nella nostra vita una serie di infrastrutture, sistemi e applicazioni su cui facciamo affidamento per svolgere le nostre attività quotidiane. L'infor- matizzazione si è introdotta in modo cosi pervasivo nelle attività umane e molti servizi sono percepiti come parte integrante dell'ambiente in cui siamo immersi. Una loro eventuale indisponibilità, interruzione o malfunzionamento ci lascia quantomeno sorpresi. Alcuni di questi sistemi e infrastrutture posso- no inoltre essere considerati critici perché la loro distruzione o un'interruzione del loro funzionamento potrebbe avere effetti catastrofici sulla sicurezza, sul- l'economia sull'ambiente o sulla salute, siano esse del singolo individuo o della società intera. In questo senso, sono ad esempio infrastrutture critiche le te- lecomunicazioni, i trasporti, il sistema elettrico, gli acquedotti, la finanza, la distribuzione del gas; sono esempi di sistemi critici il sistema di pilotaggio di un treno o di un aereo, il controllo elettronico di stabilità di un'automobile, il sistema di telecomunicazioni di un satellite. A causa della loro criticità, è quindi necessario poter garantire un certo livello di affidabilità, sicurezza o disponibilità per questo tipo di servizi. Fondamentale per un funzionamento corretto ed efficace di tali sistemi è la nostra capacità di analizzarne aspetti quantitativi relativi sia a caratteri- stiche prestazionali quali velocità di elaborazione o al-

tre misure di efficienza sia caratteristiche di sicurezza, disponibilità o affidabilità che dimostrino e ci convincano della adeguatezza dei nostri manufatti per i compiti sempre più critici e delicati per i quali li utilizziamo. Nel resto del capitolo verrà effet- tuata una trattazione sistematica della dependability, che comprende una di concetti fondamentali nella descrizione di sistemi critici quali: guasto, errore, fallimento, attributi quali affidabilità, sicurezza ed altre metriche e mezzi e tecniche per l'ottenimento della dependability.

## 2.2 La dependability

In risposta al crescente uso di calcolatori per il controllo di servizi ed attività critiche, sono state negli anni formalizzate tecniche di progettazione e di anali- si che permettano di ottenere e di garantire le necessarie qualita di un sistema dal punto di vista della adeguatezza del servizio erogato. La dependability è una delle proprietà fondamentali dei sistemi informatici insieme a funziona- lità, usabilità, performance e costo. Per fornirne una prima definizione, è necessario illustrare i concetti di servizio, utente e funzione del sistema [1].

**Definizione 1 (Servizio, Utente, Funzione).** Il servizio fornito da un sistema è il comportamento del sistema stesso, cosi come viene percepito dai suoi utenti. Un utente di un sistema è un altro sistema che interagisce attraverso l'interfaccia del servizio. La funzione di un sistema rappresenta che cosa ci attendiamo dal sistema; la descrizione della funzione di un sistema è fornita attraverso la sua specifica funzionale. Il servizio è detto corretto se realizza la funzione del sistema.♣

Possiamo ora fornire una definizione di dependability.

**Definizione 2 (Dependability).** Nella sua definizione originale, la dependability è la capacità di un sistema di fornire un servizio su cui

è possibile fare affidamento in modo giustificato. ♣ Una definizione alternativa, che stabilisce un criterio per decidere se un determinato servizio è dependable, definisce la dependability di un sistema come la capacità di evitare fallimenti che siano più frequenti e più severi del limite accettabile [1]. Questa definizione sottintende un'importante problematica: prevede infat- ti che una definizione del comportamento del sistema sia disponibile. Non sempre è semplice fornire una specifica precisa, completa e non ambigua di un sistema: infatti il comportamento corretto del sistema deve essere spesso ricavato a partire dai requisiti degli utenti, che sono solitamente impliciti e ambigui. Inoltre, per fornire una specifica completa del sistema, è necessa- rio determinare anche quali sono le condizioni ambientali (esterne) richieste affinché il sistema fornisca il servizio specificato. In altre parole, la dependability puo essere vista come una misura di quan- ta fiducia possiamo riporre in modo giustificato sul servizio erogato dal sistema stesso. Un'esposizione sistematica dei concetti relativi alla dependability con- siste di tre parti: i suoi attributi (attributes), le minacce (threats) e i mezzi (means) per ottenerla (Figura 1.1).

- Le minacce o impedimenti alla dependability. Gli impedimenti sono le cause potenziali di comportamenti non previsti.
- Gli attributi della dependability. Gli attributi ci permettono di esprimere e verificare il livello di dependability richiesto od ottenuto.
- I mezzi per ottenere la dependability. I mezzi sono le tecniche che permettono di ottenere comportamenti corretti, nonostante il verificarsi degli impedimenti.

### 2.2.1 Le Minacce: guasti, errori e fallimenti

**Definizione 3 (Guasto, Errore, Fallimento).** Si definisce guasto la causa accer- tata o ipotizzata di un errore, derivante da malfunzionamenti di componenti, interfe- renze ambientali di natura fisica, sbagli dell'operatore o da una progettazione fallace. Un errore è la parte dello stato del sistema che può causare un susseguente falli- mento; in alternativa si definisce errore la manifestazione di un guasto all'interno di un programma o di una struttura dati. Un fallimento di sistema è un evento che occorre quando un errore raggiunge l'interfaccia di servizio, alterando il servizio stes- so. Quando un sistema viola la sua specifica di servizio si dice che è avvenuto un fallimento; il fallimento è quindi una transizione da un servizio corretto a un servizio non corretto. La transizione inversa, da un servizio non corretto ad uno corretto, è detta ripristino (vedi Figura 1.2). ♣ Il guasto può rimanere dormiente per un certo periodo, fino alla sua at- tivazione. L'attivazione di un guasto porta ad un errore, che è la parte dello stato del sistema che può causare un successivo fallimento. I guasti di un sistema possono essere classificati secondo diversi punti di vista, ad esempio fisico, logico e di interazione. Un'altra suddivisione può essere fatta in base alla natura del guasto: un guasto può essere intenzionale o accidentale, mali- zioso oppure non malizioso; ed ancora in base alla persistenza dove abbiamo guasti permanenti, transienti ed intermittenti. Per una tassonomia completa si rimanda a [1]. Il fallimento di un componente si verifica quando il servizio fornito devia dalla sua specifica: si verifica nel momento in cui un errore del componente si manifesta alla sua interfaccia, e diventa quindi un guasto per il sistema. Il fallimento è quindi l'effetto, osservabile esternamente, di un errore nel siste- ma; gli errori sono in stato latente fino a che non vengono rilevati e/o non producono un fallimento. La deviazione dal servizio corretto può assumere diverse forme, che ven- gono chiamate modi di fallimento e possono venire classificati secondo la loro gravità (severity). I modi

di fallimento caratterizzano un servizio non corretto da quattro punti di vista: i) il dominio dei fallimenti, ii) la possibilità di rilevare i fallimenti, iii) la consistenza dei fallimenti, iv) le conseguenze dei fallimenti. Una completa analisi dei rischi (risk analysis, [2]) e delle modalità di fal- limento ad essi associate è necessaria per lo sviluppo di sistemi critici; tali attività sono generalmente classificate come obbligatorie negli stessi standard per la certificazione del rispetto di requisiti di dependability (ad esempio in [2]). Un sistema è formato da un insieme di componenti che interagiscono tra loro, perciò lo stato del sistema è l'insieme degli stati dei suoi componenti. Un guasto causa inizialmente un errore nello stato di uno (o più) componenti, ma il fallimento del sistema non si verifica fino a quanto l'errore non raggiunge l'in- terfaccia del servizio. La propagazione di errori può permettere ad un errore di raggiungere l'interfaccia di servizio. Questo insieme di meccanismi costi- tuisce la catena di impedimenti guasto-errore-fallimento (fault-error-failure) mostrata in Figura 1.3. La propagazione all'interno di un componente (propagazione interna) è causata dal processo di elaborazione: un errore viene successivamente trasfor- mato in altri errori. La propagazione da un componente A verso un componen- te B che riceve un servizio da A (propagazione esterna) (Figura 1.4) avviene quando un errore raggiunge l'interfaccia di servizio del componente A. A que- sto punto, il servizio che B riceve da A diventa non corretto e il fallimento di A appare a B come un guasto esterno, e si propaga come un errore all'interno di B (Figura 1.4).

# 3 Costruzione del dataset

Per addestrare una intelligenza artificiale supervisionata è necessario predisporre un dataset, da usare sia per il training sia per la fase di validation. Tensorflow, richiede per l'addestramento di una cnn, un dataset composto da un training set e da un validation set. Entrambi

a loro volta sono composti da dati di una categoria e l'altra divisi fra di loro. Nel caso del progetto, le due categorie di dati sono, immagini pulite e immagini modificate. Per la costruzione del dataset necessario al training del detector, sono state predisposte due fasi.

## 3.1 Acquisizione

La prima fase del processo di costruzione, prevede l'acquisizione di immagini pulite dal simulatore CARLA. Questo avviene grazie a un progetto di "inserisci nome del progetto github", il quale avvia n simulazioni diverse di guida autonoma. Di ciascuna di queste salva un numero fisso di immagini, acquisite dal sensore della fotocamera RGB della macchina. Nel caso di questo progetto il numero di simulazioni avviate è stato di 500, da cui per ciascuno sono state prodotte 300 immagini (800*600)

## 3.2 Sporcatura

La seconda fase del processo di costruzione, prevede la "sporcatura" delle immagini salvate nella prima fase, immagazzinandole in maniera tale da essere poi usate dall'IA. Per "sporcatura" di una immagine, è inteso l'applicazione di un effetto all'immagine che ne simuli un guasto al sensore RGB della vettura. Per fare questo è stato usato il progetto github "progetto secci", adattando le funzioni al progetto. I diversi effetti utilizzati nel progetto sono le seguenti: - - - - - differenza tra death pixel 50 e death pixel 200

# 4 Costruzione del detector

La costruzione del detector, è costituita da due fasi: