Tesi di Laurea

Sistema di riconoscimento di eventi sportivi basati su logiche temporali.

Relatori
Prof. Fabrizio Lamberti
Dr. Lia Morra

Candidato
Giuseppe Canto

Ottobre 2019

Indice

Capitolo 1

Introduzione

Questo lavoro di tesi ha come obiettivo quello di riconoscere in maniera automatica eventi di interesse in partite di calcio, utilizzando principalmente tecniche di intelligenza artificiale e machine learning, per raccogliere dati e poterli utilizzare per ragioni che possono essere: analisi statistiche sui match sportivi, analisi di performance sui giocatori o squadre di giocatori, ma anche rilevamento e analisi di tattiche di gioco.

In letteratura sono presenti numerosi esempi di strumenti ideati per il riconoscimento di eventi sportivi, ma sviluppati per lo più a partire da dati di natura visiva, cioè per esempio a partire da registrazioni di match sportivi. Pochi invece utilizzano dati posizionali di giocatori e palla, per esempio ottenuti con tecniche di computer vision o setup di telecamere. Inoltre essi, a differenza delle immagini, sono trattabili più facilmente. Questa serie di ragioni ha motivato in questo lavoro di tesi un netto interesse per il loro utilizzo.

La produzione di questi dataset con inclusi i dati posizionali richiede spesso l'installazione di array di telecamere sul campo da gioco, per la necessità di evitare il problema delle occlusioni visive dei giocatori e della palla, e pertanto ne rende costosa la generazione. Dati i costi, l'accesso a questi dataset non è mai stato trovato libero, ma a pagamento e motivato dall'interesse di natura commerciale.

Per far fronte alla mancanza di un dataset con dati posizionali, si è usato un simulatore di gioco open source che è stato opportunamente modificato per fare in modo che, in maniera del tutto automatica, fosse in grado di esportare i dati posizionali e allo stesso tempo annotare gli eventi. Per far ciò, è stato necessario intervenire sul motore di gioco e arricchirlo con una interfaccia di esportazione. In totale il dataset contiene un minutaggio pari a 500 minuti di gioco, suddiviso in 250 minuti per training set e 250 minuti per testing set, con un totale di all'incirca 8000 eventi (più di 4000 casi di passaggi, 60 casi di passaggi filtranti, più di 300 casi di cross). Questo lavoro è stato affrontato da un collega di tesi, il quale si è occupato della modifica al videogioco e di tutta la generazione del dataset. La mia parte di lavoro ha visto lo studio e lo sviluppo dell'interfaccia del videogioco per l'esportazione delle annotazione degli eventi.

Il gioco del calcio è formalizzato con regole precise che coinvolgono i giocatori, cioè gli attori, e la palla. Gli eventi di questo gioco possono pertanto essere formalizzati per mezzo di fatti e regole, e ciò ha motivato l'utilizzo di un sistema esperto basato su regole. Queste regole, inoltre, rispettano logiche temporali, che esprimono le clausole temporali che devono essere rispettate per l'occorrere degli eventi, e pertanto vanno incluse nelle loro definizioni.

Per lo sviluppo del sistema di riconoscimento si è deciso di introdurre una distinzione tra eventi semplici, detti atomici, ed eventi complessi per meglio strutturare il sistema secondo un approccio divide-et-impera. I primi sono eventi circoscritti ad un intervallo temporale e spaziale ristretto, mentre i secondi sono la combinazione di eventi semplici o di altri complessi.

Sin dalla generazione dei dati fino al loro utilizzo si è lavorato in collaborazione con altri due colleghi del gruppo di ricerca, con i quali, in maniera complementare, si è potuto dividere il lavoro. Mentre i due colleghi si sono incentrati nella generazione del dataset per mezzo di un simulatore di gioco, e nello sviluppo del primo blocco del sistema di riconoscimento, quello finalizzato agli eventi atomici, il mio lavoro si è incentrato sullo sviluppo del sistema di regole complesse, il secondo blocco del sistema di riconoscimento, in maniera che esso codificasse mediante fatti e predicati le regole del gioco del calcio.

Capitolo 2

Stato dell'arte

2.1 Revisione della letteratura

L'attività di confronto con la letteratura è il primo passo per la stesura di un lavoro che vuole migliorare i risultati per il riconoscimento di eventi sportivi, e trovare una soluzione più valida nell'approccio e nella metodologia. Qui di seguito espongo una rassegna dei documenti che si sono ritenuti più validi per il fine del progetto di tesi.
Una prima analisi va fatta sopra il concetto di evento, in quanto elemento cardine del lavoro che si andrà a discutere. Come definito in [5] è necessario un cammino a più passi per riuscire a trasformare un dato in sé in un evento significativo, d'ora in avanti evento semantico. Il processo per compiere questo cammino può variare ma ripercorre per lo più quattro passi fondamentali:

1. La raccolta del dato
2. Riconoscimento di attore e oggetto
3. Riconoscimento di azione
4. Riconoscimento di evento semantico

Man mano che da un passo si va a quello successivo la quantità di informazioni diventa sempre più condensata, e sempre di più alto livello, fino a raggiungere le conclusioni dove si ha corrispondenza con l'emissione dell'evento semantico.

Dataset di natura video sono molto diffusi in letteratura. A partire da essi si può pensare di estrapolare delle informazioni che dipendono dal contesto e dal livello di dettaglio che si prefigge. Ripercorrendo quanto specificato prima, si possono prima estrapolare le informazione di alto livello e poi quelle più di dettaglio, oppure al contrario partire dalle informazioni salienti e poi giungere ad una conoscenza specifica. In ogni caso, a partire da un flusso video, per capire se vi è interazione di attori con altri attori o di attori con oggetti è necessario prima rilevare gli attori e gli oggetti, quindi prefissare un approccio bottom-up che, partendo dalle informazioni più di basso livello mira a riconoscere l'evento solo se si è stati in grado di rilevare gli attori, gli oggetti e la loro interazione.

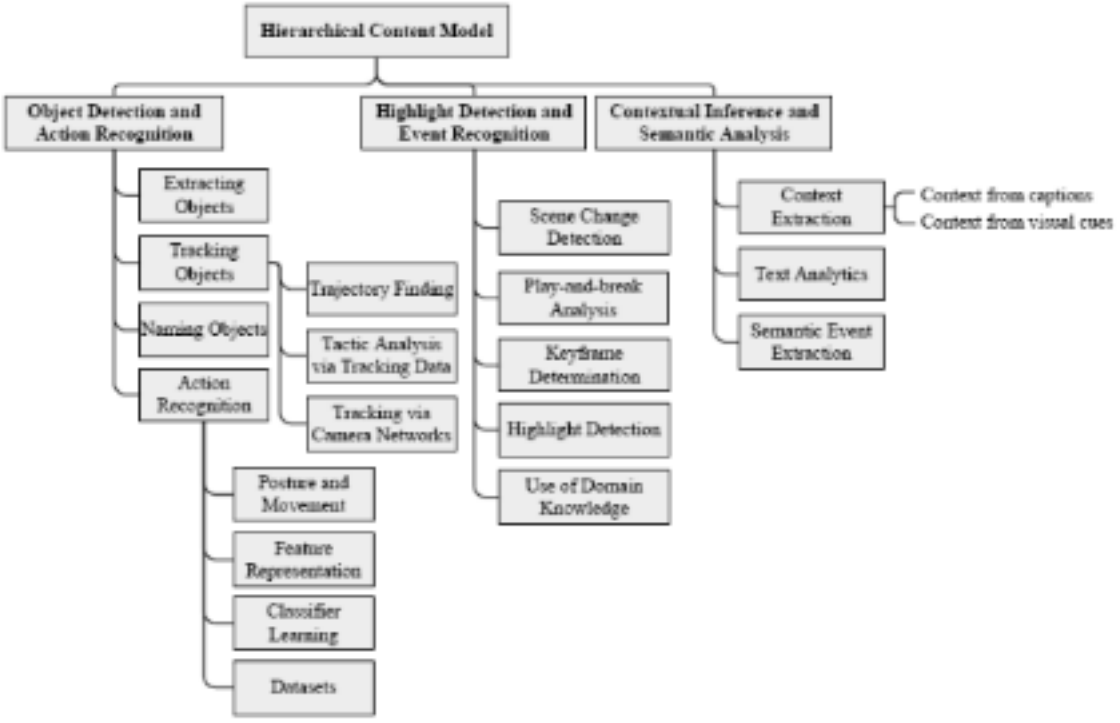


Figura 1 - Possibili aree di ricerca.

Alcuni algoritmi basati su riconoscimento di oggetti e riconoscimento di azioni hanno però dei limiti. A partire da un flusso video è possibile che gli attori o gli oggetti vengano occlusi, soprattutto laddove ci sono molti attori e/o oggetti. Quindi è necessaria una scansione tridimensionale del campo di gioco. Altra possibile alternativa è quella di riconoscere il testo o i numeri rappresentati sulle maglie dei giocatori ma anche questa opzione presenta dei limiti: è possibile che in alcune inquadrature queste non siano visibili.

Altri lavori [16] che partono invece da dati posizionali, ma basati su machine learning, mostrano che genere di features possono essere utili per il riconoscimento di eventi. La classificazione è basata sul cambiamento assunto da queste features costruite sui dati.

Esempi sono:

- Direzione
- Velocità
- Accelerazione
- Picco di accelerazione
- Cambio di direzione
- Distanza dall'obiettivo

Altro elemento decisivo è stato fornito nel documento [6] dove si descrive che per il riconoscimento di eventi sportivi è efficace procedere con sistemi che presentano una base conoscenza a priori del dominio, che ha motivato l'utilizzo di sistemi esperti per questo lavoro di ricerca. In particolare, così come sono presenti regole nel gioco sportivo è possibile ricostruire un sistema basato su regole che descrive le prime e consente la rilevazione degli eventi di interesse. In particolare è descritto che, avendo una base di conoscenza circa il dominio in cui si sta lavorando è possibile usare dei sistemi ad inferenza per il riconoscimento e l'emissione dei tipi di eventi. La pubblicazione riporta come base di dati il tracciamento delle posizioni degli attori e degli enti nel tempo. Questi ultimi alimentano poi un sistema basato su regole per la loro effettiva elaborazione. Uno schema esemplificativo è riportato in figura 2.

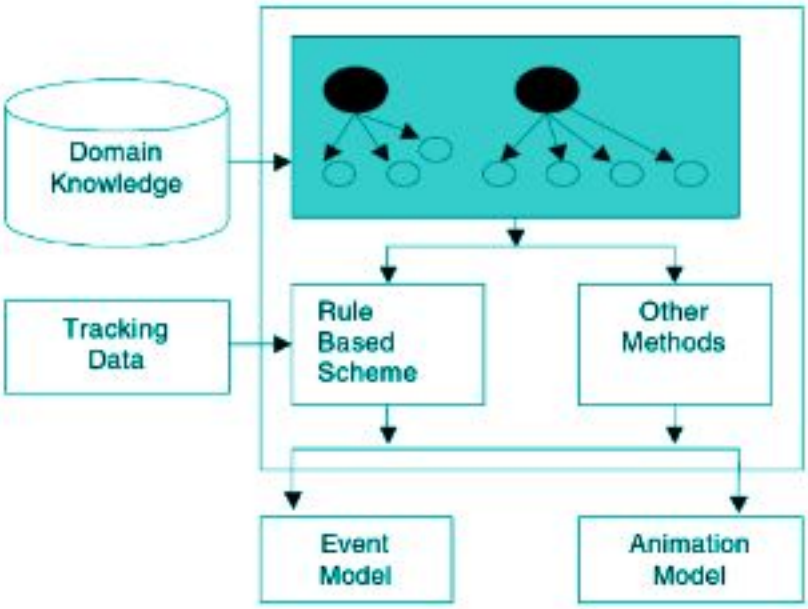


Figura 2 - Schema rappresentativo del sistema proposto in [6].

La letteratura riporta comunque l'utilizzo di altre metodologie come per esempio algoritmi di machine learning che affrontano il problema del riconoscimento dei dati a partire da dati posizionali, dapprima con la raccolta di *features*, successivamente con la l'apprendimento di un modello. La peculiarità è che questo modo di affrontare il problema va bene se si decide di usare tanti modelli quanti sono gli eventi da riconoscere perché ogni evento ha caratteristiche semantiche diverse e richiede features pressoché differenti. Considerato che i giochi sportivi, e soprattutto il calcio, il

gioco preso in esame, sono molto dinamici e in essi si verificano eventi imprevedibili, è necessario avere numerosi modelli.
Un esempio su come formalizzare un evento è invece proposto in [7]. L'idea di base è creare una serie di eventi semplici, che costituiscono i mattoni fondamentali con cui legare i successivi in una serie di rapporti logici. Gli eventi complessi possono essere desunti grazie alla combinazione dei primi, attraverso una relazione descritta mediante operatori logici. Pertanto è stata valutata la scelta di un motore di processamento di eventi, ricaduta poi in Etalis, che permetta di implementare delle regole con operatori logici sulla base dei suddetti eventi semplici. Il lavoro fatto a partire dal paper è stato dapprima incentrato nella definizione formale degli eventi semplice e degli eventi complessi, e poi incentrato proprio sulla scrittura del sistema di regole, codificando le complesse regole del gioco in regole logiche con clausole spazio-temporali.

In [19] viene affrontato il problema di combinare sistemi di logiche temporali a sistemi di logiche spaziali. Infatti, mentre sono diffusi in letteratura casi di studio che fanno uso di sistemi a logiche temporali per modellare problemi del mondo reale, pochi si trovano ad affrontare casi che possono essere modellati con entrambe .logiche temporali e spaziali. Esso fa uso di relazioni spaziali per muoversi tra oggetti, e riconoscere l'interazione tra gli attori con gli oggetti.

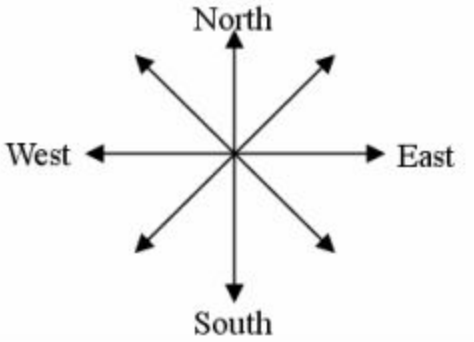


Figura 3 - Relazioni spaziali.

Questo lavoro è di particolare interesse perché sarà necessaria una certa formalizzazione per il riconoscimento di eventi che includono clausole di natura spaziale.

Nel paper [20] si parla di logiche temporali e si contraddistingue principalmente la distinzione tra rappresentazione a istanti di tempo nei confronti di quella a intervalli di tempo. Con rappresentazione a istanti di tempo si fa riferimento a una descrizione del tempo che avviene per punti, dove il passare del tempo è dato dal susseguirsi di questi punti e un approccio di rappresentazione di questo tipo è utile per quei casi in cui si

vuole fare ordinamento temporale di eventi istantanei. Per rappresentazione a intervalli di tempo invece si fa riferimento ad una modellazione ontologica in cui il problema contiene eventi che hanno una durata finita di tempo.

In [21] invece viene mostrata la differenza tra modelli ontologici spaziali per la descrizione di entità ed oggetti nello spazio. Si confrontano:

- Modelli spaziali basati su punti
- Modelli spaziali basati su regioni

Per i primi lo spazio è identificato da una collezione di punti dello spazio che possono essere legati tra di loro per mezzo di operatori spaziali e sono definiti all'interno di uno spazio topologico. Quindi un oggetto è definito per mezzo dell'insieme di punti che esso occupa. Per i secondi, gli oggetti sono rappresentati come elementi di forma variabile. Combinando una rappresentazione spaziale e temporale si ottiene una descrizione ontologica molto ricca.

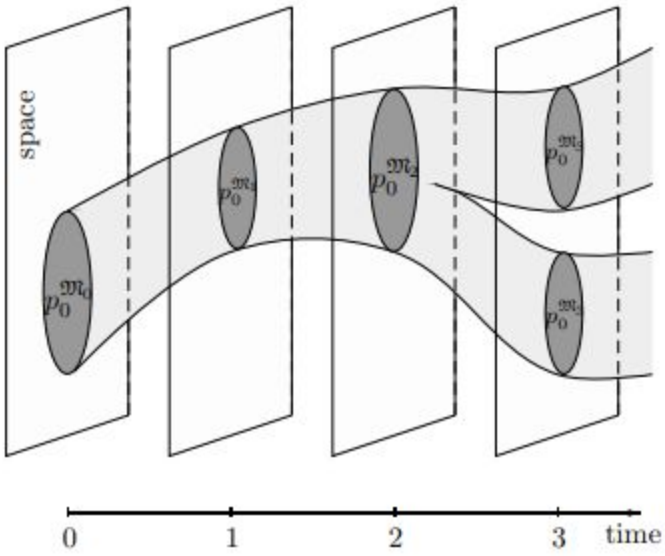


Figura 4 - Oggetto di forma circolare definito all'interno di un certo spazio, che si muove nel tempo in maniera lineare.

All'interno dello spazio è poi possibile definire una relazioni topologiche di distanza tra i vari oggetti. Questi saranno utili per formalizzare clausole logiche di natura spaziale che coinvolgono attori ed oggetti.

Capitolo 3

Logiche temporali

In un modo o nell'altro, ogni area dell'intelligenza artificiale (AI) ha a che fare con il tempo. I sistemi di diagnosi medica ragionano sul momento in cui un virus ha infettato il sistema sanguigno. I sistemi di risoluzione dei problemi dei dispositivi osservano quanto tempo impiega un condensatore a saturare. Nella programmazione automatica, il momento in cui una variabile diventa vincolata è importante. Nella pianificazione dei robot, si vuole raggiungere un obiettivo prima di un altro per rispettare le scadenze e così via. Nel contesto del lavoro di tesi, si vuole descrivere che successione di eventi porta l'occorrere dell'azione sportiva.
Possiamo considerare le logiche temporali come un linguaggio per codificare una certa conoscenza temporale. Per fare ciò e per caratterizzare questo linguaggio si può scegliere tra più alternative: considerare il tempo come discreto o continuo, basato su istanti o basato su intervalli, deterministico o no, lineare o circolare.
Nella maggior parte dei casi di ragionamento temporale, può bastare che ci siano istanti di tempo come entità temporali ontologiche di base. Tuttavia, può succedere che gli istanti di tempo non sono adatti a ragionare adeguatamente su eventi del mondo reale, che hanno un'intrinseca durata. Per questa ragione può essere necessario scegliere un modello basati su intervalli. Si noti che una volta deciso di optare per un modello basato su intervalli, si può sempre considerare un solo estremo e far degenerare l'intervallo in istante.

Modelli basati su istanti temporali

Nei modelli basati su istante l'unità primitiva è l'istante di tempo, e il fluire del tempo lo si rappresenta come un insieme di istanti di tempo con una relazione di precedenza su essi.

Figura 5 - Esempio di modello basato su istanti e a tempo lineare.

Semantica delle logiche temporali per modelli a istanti temporali

Si parte definendo un *temporal frame,* all'interno del quale il tempo fluisce. Poi si definisce un *temporal model* per un insieme di proposizioni atomiche. A questo punto è necessario verificare le proposizioni, ossia bisogna assegnare un valore di verità a ciascuna delle proposizioni atomiche per ogni istante di tempo, e a quel punto in maniera induttiva si dice che la formula è valida se è vera in tutti gli istanti di tempo, altrimenti è detta soddisfacibile se è vera in qualche istante di tempo.

Modelli basati su intervallo

Frasi come "la scorsa notte Alice ha pianto mentre scriveva la lettera, e poi si è calmata" possono essere rappresentate solo con modelli basati su intervallo. Essi sono ontologicamente più ricchi, dal momento che ci possono essere più relazioni tra intervalli di tempo che tra istanti di tempo. Rappresentate in fig. 3 vi sono alcune possibili relazioni tra intervalli e successiva formalizzazione.

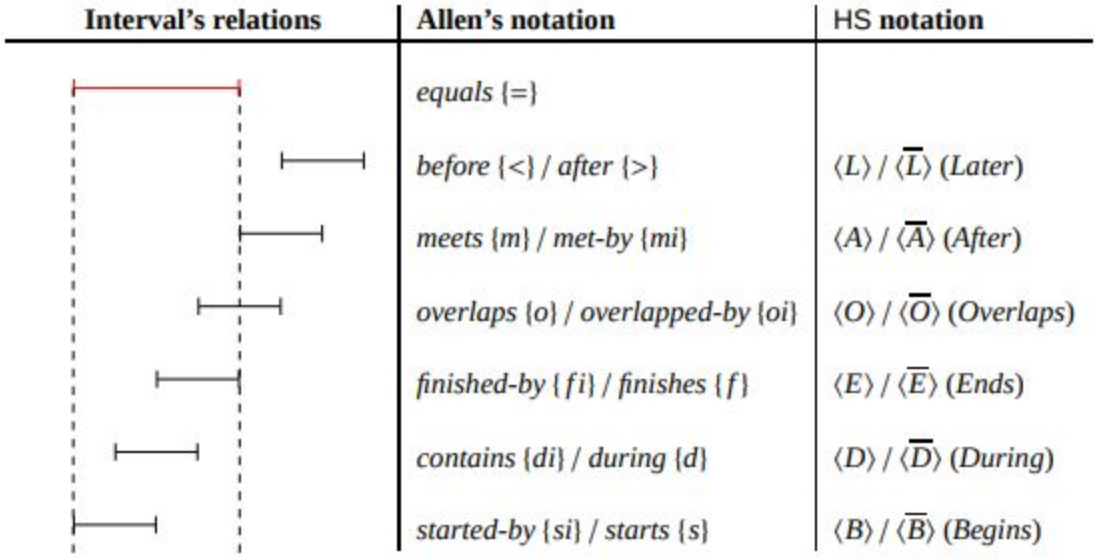


Figura 6 - Relazioni tra coppie di intervalli, secondo la notazione di Allen, e Halpern-Shoham.

Definizione di intervallo

Un intervallo in D è una coppia ordinata [d0, d1] tale che d0, d1 ∈ D e d0 ⩽ d1. Un punto d appartiene a un intervallo [d0, d1] se d0 ⩽ d ⩽ d1. Se d0 <d1, allora [d0, d1] viene chiamato intervallo stretto. Le strutture di intervallo considerate qui siano lineari, ovvero ogni due punti sono comparabili. Questa restrizione di solito può essere rilassata senza complicazioni per gli ordinamenti parziali con la proprietà dell'intervallo lineare, ossia in ogni ordinamento parziale, ogni intervallo è lineare.

Capitolo 4

Progettazione

4.1 Formalizzare il gioco del calcio

Il calcio è uno sport di squadra giocato con un pallone su un campo di gioco rettangolare, con due porte. Le squadre sono composte da 11 giocatori. Dieci di essi possono toccare il pallone solo con i piedi, il corpo e la testa; uno solo posto a difesa della porta (e perciò detto "portiere"), può toccare il pallone anche con mani e braccia, solamente se il pallone si trova in area di rigore. L'obiettivo del gioco è quello di segnare più punti (detti gol o reti) della squadra avversaria, facendo passare il pallone fra i pali della porta avversaria. La durata di una partita è di 90 minuti, divisi in due tempi da 45' ciascuno più un eventuale recupero.

Definizione campo di gioco

Per prima cosa è necessario definire le dimensioni del campo. Il regolamento del gioco non prevede delle dimensioni fisse per il campo ma dei range possibili. Nel caso del lavoro di tesi, in accordo con l'appena citato regolamento, si è fatto riferimento alle seguenti estensioni:

- 70 m di estensione in altezza
- 110 m di estensione in larghezza

Il punto di riferimento iniziale è il centro del campo con coordinate (0,0) e i valori degli assi variano da -55 a +55 per l'asse delle ascisse x, e da -35 a +35 per l'asse delle ordinate y.

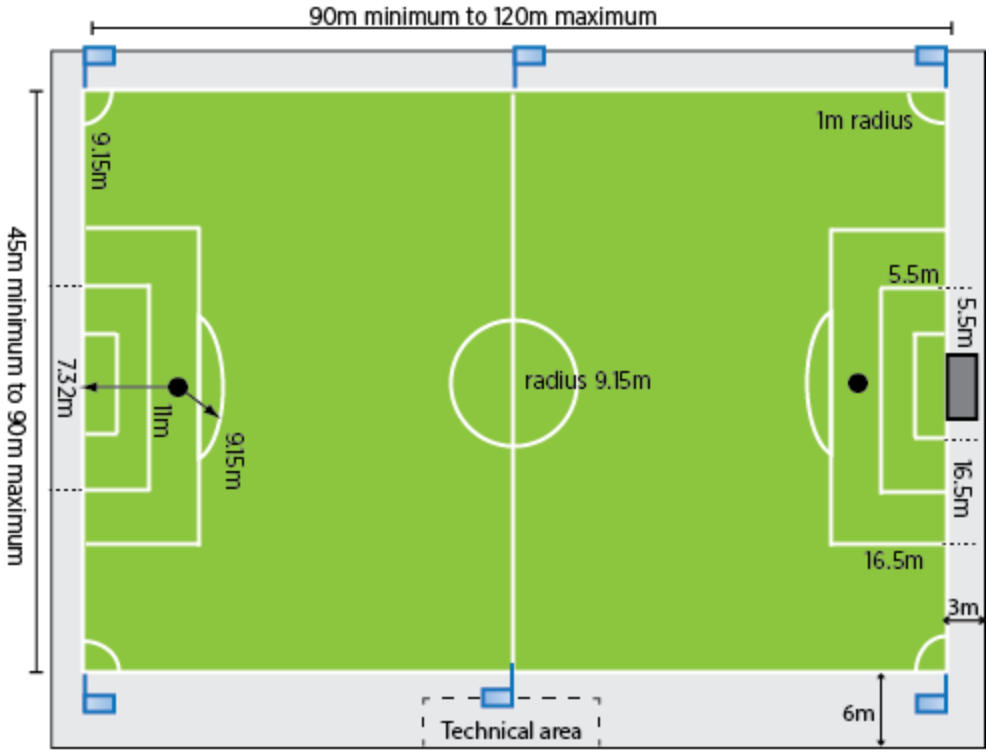


Figura 7 - Esempio di campo da calcio e sue dimensioni. Le immagini provengono dal Dipartimento dello Sport e Ricreazione, Stato dell'Australia dell'Ovest.

Assume un interesse strategico la suddivisione del terreno di gioco in macroaree, che permetterà di descrivere in maniera precisa le regioni entro cui una dinamica di evento viene consumata. Le regioni di interesse sono: fasce laterali (side), interno campo (inner-pitch), area di rigore (penalty area), area di porta (goal area).

I dati di particolare interesse sono:

- La regione delle fascie detta Side si estende da 0 a 110 sull'asse delle ascisse, mentre su quello delle ordinate da 0 a 19.5 e poi da +19.5 a +36.

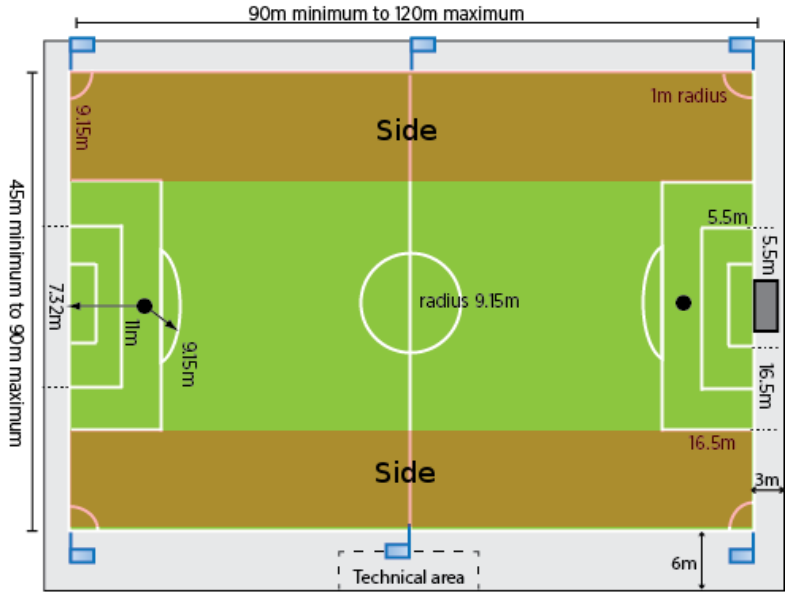


Figura 8 - Fasce laterali del terreno di gioco.

- La regione penalty-area si estende da +15.85 a +56.15 sull'asse delle ordinate e sull'asse delle ascisse da 0 a +16.5 e da +93.5.5 a +110.

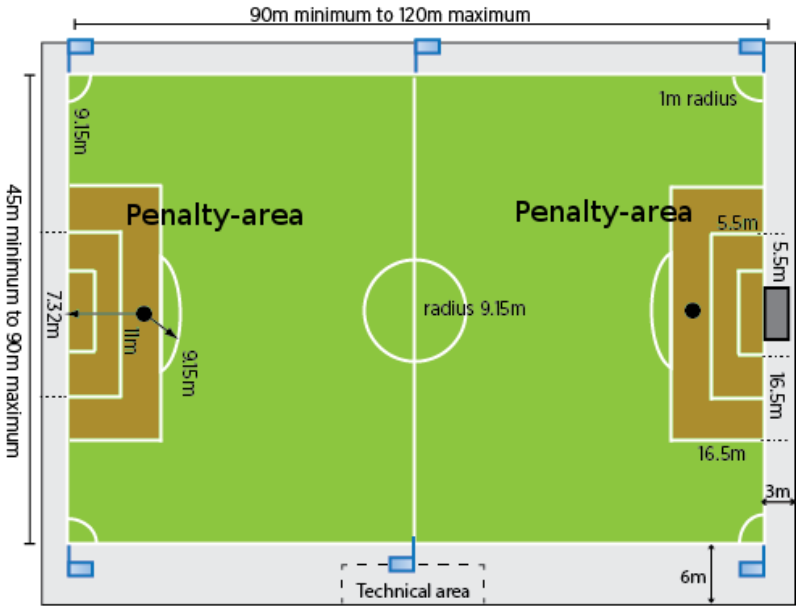


Figura 9 - Area di rigore del terreno di gioco.

- La regione goal-area si estende da +26.85 a +45.15 sull'asse delle ordinate, mentre si estense sull'asse delle ascisse da 0 a +5.5 e da +104.5 a +110.

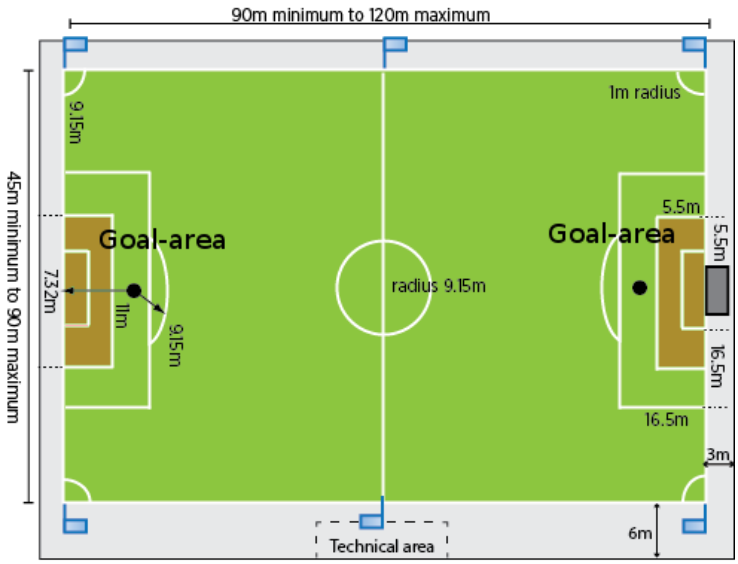


Figura 10 - Aree di porta del terreno di gioco.

- La regione dell'inner-pitch si estende da +15.85 a +56.15 nell'asse delle ordinate, mentre si estende sull'asse delle ascisse da 0 a +27.5 e da +82.5 a +110.

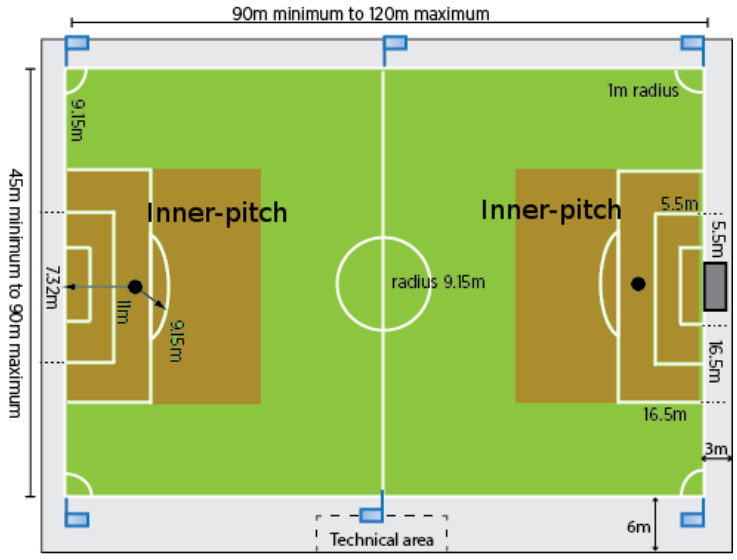


Figura 11 - Interno campo del terreno di gioco

4.2 Eventi calcistici

Nel gioco del calcio, così come in tutti gli sport, è possibile determinare una serie di eventi semantici che da una parte esprimono le regole del gioco, dall'altra, se rilevati, consentono a posteriori di fare delle analisi per fini tattici o statistici. Gli eventi di interesse, o di *tassonomia*, considerati in questo lavoro di tesi sono i seguenti, mostrati in Tabella 1.

Eventi		Descrizione
1 - Passaggi		
1.1	Effettuati	Giocatore che calcia la palla e consente ad un compagno di squadra di ottenere il possesso palla.
1.2	Filtranti effettuati	Giocatore che calcia la palla e consente ad un compagno di squadra di ottenere il possesso palla riuscendo a superare l'ultimo difensore della squadra avversaria.
1.3	Ricevuti	Giocatore che riceve la palla da un compagno di squadra per mezzo di un calcio palla.
1.4	Filtranti ricevuti	Giocatore che riceve la palla oltre la linea dell'ultimo difensore della squadra avversaria, per mezzo di un calcio palla.
1.5	Ricevuti che sono sfociati in goal	Giocatore che riceve la palla per mezzo di un calcio palla e che, dopo aver effettuato un controllo palla, calcia in porta e fa goal.
1.6	Filtranti ricevuti che sono sfociati in goal	Giocatore che riceve la palla oltre la linea dell'ultimo difensore della squadra avversaria, per mezzo di un calcio palla, e dopo aver effettuato un controllo palla, calcia in porta e fa goal.
2- Cross		
2.1	Effettuati (falliti)	Giocatore che calcia la palla da una zona laterale del terreno di gioco e spinge la palla nella zona di interno campo.
2.2	Corretti effettuati	Giocatore che calcia la palla da una zona laterale del terreno di gioco e spinge la palla nella zona di interno campo, consentendo ad un compagno di squadra di ottenerne Il possesso.
2.3	Ricevuti	Giocatore che riceve la palla nella zona di interno campo per mezzo di un calcio palla avvenuto nella zona laterale del terreno di gioco.

2.4	Effettuati che sono sfociati in goal	Giocatore che calcia la palla da una zona laterale del terreno di gioco e spinge la palla nella zona di interno campo, consentendo ad un compagno di squadra di ottenerne Il possesso, di tirare in porta e fare goal.
4- Contrasto		
4.1	Fatto (fallito)	Tentativo di togliere il controllo palla ad un giocatore della squadra avversaria.
4.2	Vinto	Tentativo avvenuto con successo di togliere il controllo palla ad un giocatore della squadra avversaria.
6- Tiri		
6.1	Totali	Somma dei tiri avvenuti durante la partita.
6.2	Tiri goal	Giocatore che calcia la palla in direzione porta e riesce a fare goal.
6.3	Tiri fuori	Giocatore che calcia la palla in direzione porta ma la palla va fuori, ai lati della porta o in altro, sopra la traversa.
6.4	Tiri parati	Giocatore che calcia la palla in direzione porta ma il portiere ne blocca il tentativo di rete.
7- Portieri		
7.1	Parate su tiri dentro area piccola	Portiere che effettua parata, da un tiro avvenuto dentro l'area di porta.
7.2	Parate su tiri dentro area di rigore	Portiere che effettua parata, da un tiro avvenuto dentro la penalty area.
7.3	Parate su tiri da fuori area	Portiere che effettua parata, da un tiro avvenuto al di fuori della penalty area.
7.4	Rigori parati	Portiere che effettua parata, da un tiro avvenuto dal dischetto.
7.5	Punizioni parate	Portiere che effettua parata, da un tiro avvenuto per fallo.

Tabella 1 - Eventi di interesse, o di *tassonomia*.

4.3 Dataset disponibili

Visti e considerati gli obiettivi di ricerca precedentemente definiti, è importante ricercare insiemi di dati che abbiano o dati visivi, immagini o video, o dati di tracciamento delle posizioni dei giocatori e della palla. I dati visivi permettono, ad un osservatore umano o artificiale, di capire quale è la dinamica di gioco avvenuta, mentre i dati di tracciamento delle posizioni sono l'approssimazione minima che descrive l'andamento del gioco, e pertanto rappresentano una forma concisa ed ottimizzata del gioco condotto. I dati visivi possono essere utili per un operatore esperto a comprendere quali azioni del gioco sono avvenute durante una partita e a supportare pertanto l'attività di verifica, una volta ottenuti i risultati. Oppure possono essere utili se si vuole indagare l'utilizzo di sistemi che sfruttano il machine learning che prendono in ingresso immagini e che si prestano bene alla loro elaborazione per il fine del riconoscimento di forme e figure. Tuttavia questi sistemi richiedono una collezione molto ampia di dati e i tempi di training per un modello che ne fa uso sarebbero elevati. L'utilizzo invece di dati posizionali garantisce un'elaborazione più pratica, mantenibile in termini di consumo di memoria e sostenibile come computo di elaborazione. Inoltre, in letteratura, molte soluzioni sfruttano dataset basati su immagini e pochi quelli basati su posizioni: questo rende interessante esplorazione questa strada. L'attività di ricerca condotta per il fine di trovare i suddetti insiemi di dati ha prodotto una lista di risorse che sono state attentamente scrutate e valutate sotto una serie di aspetti quali: licenze, quantità di dati, e qualità semantica dei dati.

Insieme di dati Alfheim
Questo insieme di dati [10] presenta sia riprese video effettuate con delle telecamere sia dati posizionali dei giocatori che sono stati raccolti mediante l'utilizzo di sensori indossati dai giocatori durante la partita ed un insieme di antenne radio posizionate attorno al campo di gioco. I dati però risultano poco utilizzabili per alcune ragioni: innanzitutto i dati posizionali dei giocatori si riferiscono ad una sola squadra in campo, il chè rende impossibile ricostruire azioni nella quale intervengono giocatori di squadre opposte, e in secondo luogo mancano della posizione della palla, elemento chiave per la corretta ricostruzione della dinamica di gioco.

Insieme di dati Issia CNR
Questo insieme di dati [12] comprende sia registrazioni video di una partita sia i dati posizionali dei giocatori, dell'arbitro e della palla. Il problema nell'utilizzare questi dati risiede nel fatto che sono ripresi solo due minuti di gioco e gli eventi rappresentati sono solamente alcuni passaggi.

Insieme di dati KTH

Questo insieme di dati [14] fornisce immagini che rappresentano una porzione limitata di una partita, essi rappresentano poche azioni, e sono principalmente focalizzati su determinate azioni. In compendio alle immagini ci sono sequenze scheletriche che rappresentano in maniera concisa i corpi dei giocatori in uno spazio 3d, lo spazio tridimensionale del campo di gioco. Sono utili se si vuole esplorare l'ambito di riconoscimento di azioni sportiva con dati basati su sequenze scheletriche. Argomento non di studio nel presente lavoro.

Insieme di dati Magglingen

I dati [13] si riferiscono solamente alla componente posizionale dei giocatori e della palla, ma sono privi di immagini o rappresentazioni video. Questa condizione rende particolarmente complicato il suo utilizzo perché non si è in grado di verificare quali eventi sono avvenuti effettivamente.

Insieme di dati SoccerNet

E' un insieme di dati [11] che contiene registrazioni video di partite di calcio di diverse leghe che sono state trasmesse in programmi televisivi, ma che mancano di dati posizionali dei giocatori e della palla. Questo insieme di dati è utile se si vuole sperimentare il riconoscimento dei giocatori e della palla ma non è utilizzabile per lo scopo di questo lavoro di tesi.

4.4 Generazione di un insieme di dati sintetico

L'effettiva mancanza di un insieme di dati che contenga sia immagini rappresentanti il gioco avvenuto, sia dati di tracciamento delle posizioni dei giocatori e dalla palla, ha motivato la scelta di costruire un insieme di dati sintetico che si avvicini quanto più possibile ad un incontro sportivo reale. La necessità dunque di avere dei dati in forma completa, con immagini e posizioni, ha spinto la ricerca verso un simulatore di gioco, uno strumento che rendesse possibile estrarne dati di movimento dei giocatori e della palla, e al contempo poterne registrare il videogioco. La ricerca condotta in questo verso ha prima di tutto tenuto in considerazione alcune soluzioni di natura commerciale, considerando i titoli più famosi e anche più vicini alla realtà, poi in seguito ha considerato soluzioni con codice sorgente aperto, liberamente modificabile, ma che si distaccano maggiormente dalla verosimiglianza di una registrazione sportiva televisiva. Dapprima infatti si è cercato di mantenere un'elevata qualità di dati visivi e di movimento, andando a ispezionare tra le funzioni di gioco dei titoli sopracitati, se fossero presenti dei meccanismi per esportare i dati ricercati. Tuttavia, le soluzioni commerciali studiate non permettevano operazioni di questo genere. Dunque si sono valutate soluzioni con codice sorgente aperto e valutato la possibilità di modificare il codice sorgente secondo criteri di complessità, estensibilità, e valutando soprattutto la qualità dei dati che si sarebbero andati ad esportare.

- **Eat The Whistle**: Open source remake of a classic soccer game.
- **Yoda Soccer**: Open source remake of Sensible Soccer.
- **Football Manager**: Open source remake of the first Football Manager (1982)
- **Gameplay Football**: Open source 3D soccer game, similar to early 2000s FIFA game, mostrato in figura 7.

La scelta è infine ricaduta nel simulatore di gioco *Gameplay Football*, principalmente perché è quello la cui grafica di gioco si avvicina di più alla registrazione di un incontro sportivo trasmesso televisivamente, e in secondo luogo perché integrava già da sé una prima forma di esportazione dei dati di movimento dei giocatori sotto forma di registro. Al simulatore di gioco è stata aggiunta un'interfaccia applicativa che ha permesso di esportare in forma automatizzata i dati richiesti.

Figura 12 - Esempio di dettaglio del videogioco *Gameplay Football* [15].

4.5 Sistema di riconoscimento

Un sistema di riconoscimento deve rispettare due attributi fondamentali: l'identificazione e la verifica. Nel caso del lavoro di tesi si tratta di andare a identificare quale evento è occorso in un certo momento della partita, e verificare rispetto ad un dataset della verità la sua effettiva presenza. Ognuno di questi due aspetti va considerato e studiato.
Per l'identificazione degli eventi sono state valutate due soluzioni differenti in approccio: una basata su machine learning, la seconda su un sistema di regole. Entrambe le soluzioni sono ampiamente usate per fare inferenza a partire da un insieme di dati e per capire quale usare è stata condotta un'analisi per valutare i loro punti di forza e debolezza.
I sistemi di regole sono tipicamente costruiti grazie alla conoscenza degli esperti di dominio i quali si occupano di incorporarla all'interno nel sistema, sotto forma di fatti e regole. Includere esperti di dominio nella scrittura delle regole fa sì che questi sistemi vengano chiamati *sistemi esperti*. Uno dei fattori positivi nella scelta di questi sistemi è la facilità sulla composizione delle regole. Esse riflettono lo schema If-Then, e sono quindi di immediata comprensione e implementazione ma, oltretutto, sono estensibili su due fronti: è possibile aggiungere nuove clausole aggiuntive alle regole scritte in passato senza dover riscrivere l'intero insieme di regole, oppure è possibile facilmente aggiungere un nuova regola per ampliare il precedente sistema. Un aspetto negativo è che inizialmente una regola può essere composta da poche clausole ma poi, dopo vari arricchimenti e affinamenti successivi, si raggiunge uno stadio che risulta di difficile comprensione per i non esperti. Inoltre, questa soluzione, non è valida per quegli scenari dinamici che cambiano troppo spesso, dove le regole andrebbero modificate molto velocemente. Un punto a favore sull'utilizzo di queste tecniche è sicuramente il fatto di non dover scrivere un sistema complesso di regole che richiede una conoscenza esperta di dominio, ma essendo basate su modelli, esse possono essere considerate come delle scatole nere, e ciò garantisce un livello di astrazione tale da semplificare il caso di studio. Alcuni modelli sono molto semplici e intuitivi da spiegare, i sistemi di decisione basati su alberi, per esempio. Essi permettono di visualizzare il flusso delle decisioni prese. Un punto a sfavore, invece, è il fatto di dover allenare il modello, e ciò richiede molti dati e parecchia computazione. Una netta mancanza di un buon dataset è un fattore determinante per la scelta o meno di una soluzione così. La parte complicata di allenare un modello di classificazione è che richiede moltissimi dati per l'addestramento.

La scelta è ricaduta sul sistema di regole per i vantaggi descritti sopra e per la mancanza di un dataset. Inoltre, essendo le regole del gioco del calcio già definite e disponibili, assume un carattere di coerenza costruire un sistema esperto che le implementi.
Per il procedimento di verifica degli eventi riconosciuti, si è sviluppato un modulo software chiamato *validator* che si occupa di trovare l'abbinamento tra l'evento realmente occorso e quello riconosciuto.
Per la progettazione effettiva del sistema di riconoscimento va tenuto in considerazione il dato sopra cui costruire l'architettura software, e la modalità con cui processare questo dato. Per questo motivo è stato necessario formalizzare eventi, ricercare una sintassi per la loro descrizione, e anche introdurre una suddivisione degli eventi in atomici e complessi per garantire un disaccoppiamento metodologico. Si rimanda alla lettura del paragrafo 5.2 per questi dettagli.

Dati in ingresso

Il dataset a disposizione, ottenuto grazie al simulatore di gioco, come già specificato al capitolo 4, fornisce dati di tracciamento delle posizioni dei giocatori e della palla. L'obiettivo del sistema di riconoscimento sarà pertanto quello di trasformare questi dati posizionali in un evento sportivo, portando il "dato crudo" ad un livello di astrazione superiore.

Figura 13 - Schema di alto livello del sistema di riconoscimento.

4.6 Introduzione di eventi atomici e complessi

Tenendo in considerazione gli eventi di interesse prefissati per questo lavoro di tesi sono stati prima formalizzati i nomi e le nomenclature per ciascuno di essi, poi si è riconosciuta l'evidenza che alcuni di essi avevano punti in comune, ed è scaturita così l'idea di formalizzare degli eventi più semplici e fondamentali, sulle quali costruire gli eventi di interesse (menzionati nel paragrafo 4.2) come combinazione di essi. Dopo una fase di valutazione si è deciso di formalizzare gli eventi semplici con una specifica sintassi e si è deciso di nominarli *atomici*, in quanto circoscritti ad un'area spaziale del terreno di gioco e ad un ristretto intervallo temporale, da cui appunto prendono il nome.

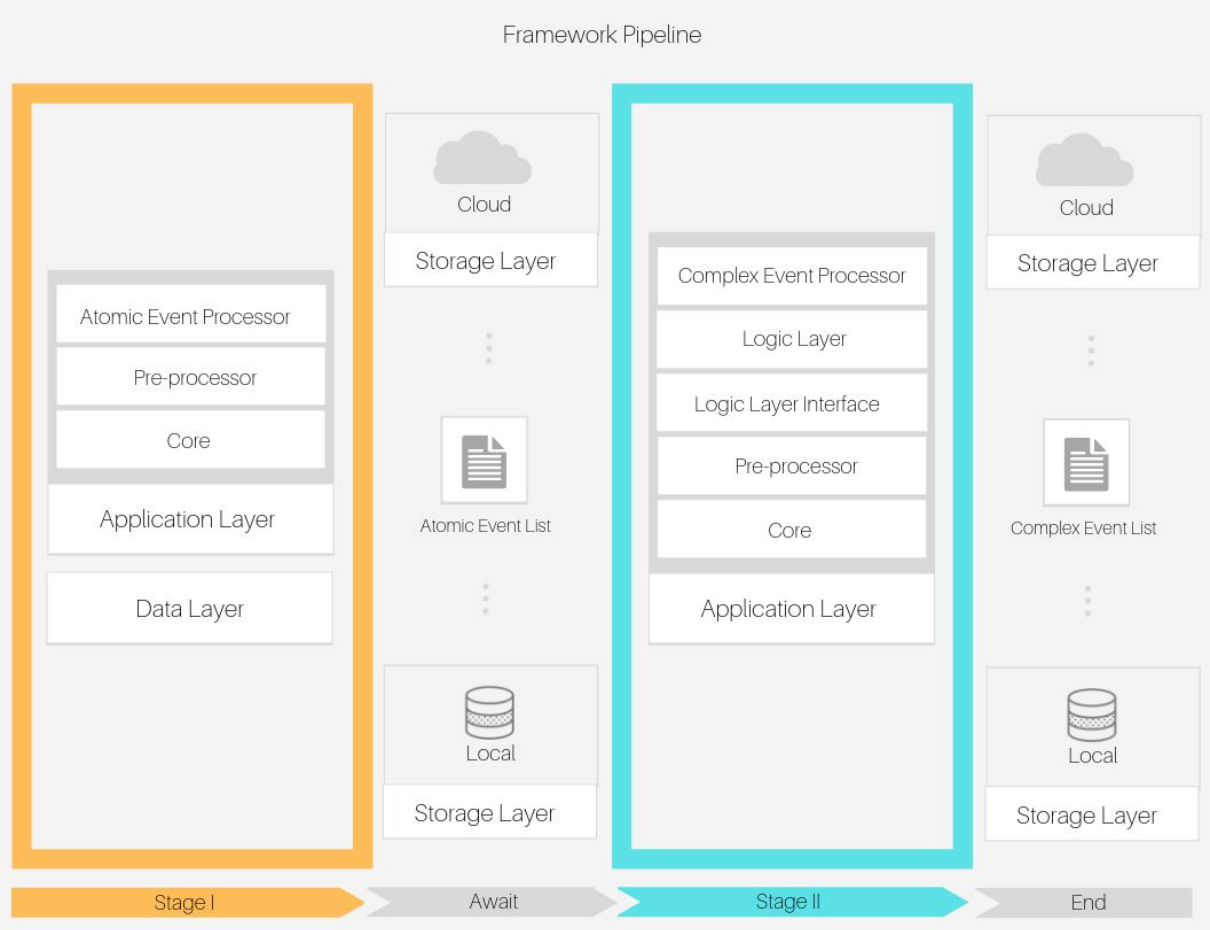


Figura 14 - Sistema di riconoscimento, diviso per atomici e complessi.

Gli eventi atomici, ispirati al lavoro descritto in [7], costituiscono la base per il sistema di inferenze dei complessi. Essi sono propedeutici al riconoscimento degli eventi

complessi. Gli eventi atomici hanno un durata istantanea, e si verificano in un'area ristretta del terreno di gioco. Gli eventi complessi invece, per definizione, si verificano in un'area estesa o molto estesa del campo di gioco e hanno un'intrinseca durata, ossia non sono istantanei. Il riconoscimento degli eventi complessi avviene per combinazione di eventi semplici, o di eventi semplici con altri eventi complessi, o per combinazione di eventi complessi. Alla luce di questa suddivisione si è ideato un framework software che segue lo schema di progettazione a pipeline e che in totale espleta la funzione di riconoscitore. Più in dettaglio, il sistema per atomici lavora sul piano della *percezione*, mentre il sistema per complessi su quello del *ragionamento*. Essi in totale operano su due piani diversi, ma sono complementari.

Formalizzazione eventi

Gli eventi atomici sono tutti formalizzati secondo la seguente sintassi:

```
<IDevento,NomeEvento,ListaDi<Attore,nomenclaturaAttore,tempo>>
```

Gli eventi complessi, invece, sono stati formalizzati secondo la seguente sintassi:

```
<IDevento, NomeEvento, T=<Evento1> THEN <Evento2> THEN . . . >
```

4.7 Sistema di riconoscimento per atomici

Il sistema di riconoscimento per eventi atomici ha come obiettivo principale riconoscere i seguenti eventi.

Evento emesso
KickingTheBall
BallPossession
Tackle
BallDeflection
Goal
Penalty
Foul
Offside

Tabella 3 - Lista eventi atomici emessi.

L'architettura di questo modulo software comprende una successione di operazioni che vengono eseguite da sottomoduli interamente dedicati a svolgere funzioni specifiche: l'estrazione delle caratteristiche dai dati in ingresso, il riconoscimento degli eventi atomici a partire dalle caratteristiche, e infine la memorizzazione in maniera persistente del dato.

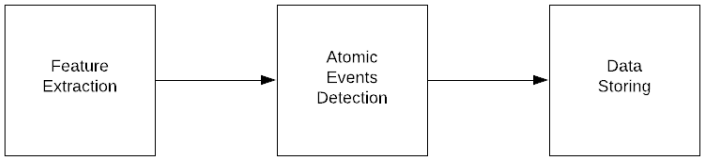


Figura 15 - Schema architetturale di massima del sistema per atomici.

4.8 Sistema di riconoscimento per complessi

Il sistema di riconoscimento per eventi complessi ha come obiettivo quello di riconoscere i seguenti eventi.

Eventi di tassonomia	Eventi emessi
Passaggi effettuati	Pass
Passaggi ricevuti	Pass
Passaggi ricevuti che sono sfociati in goal	PassThenGoal
Passaggi filtranti effettuati	FilteringPass
Passaggi filtranti ricevuti che sono sfociati in goal	FilteringPassThenGoal
Cross corretti effettuati	Cross
Cross effettuati (falliti)	Cross
Cross effettuati che sono sfociati in goal	Cross
Cross assist	CrossThenGoal
Tiri totali	Shot
Tiri fuori	ShotOut
Tiri goal	ShotThenGoal
Parate su tiri da fuori area	SavedShot
Parate su tiri dentro area di rigore	SavedShot
Parate su tiri dentro are piccola	SavedShot
Rigori parati	SavedShot
Punizioni parate	SavedShot
Contrasto vinto	Tackle
Contrasto fatto (fallito)	Tackle

Tabella 4 - Lista eventi complessi emessi, rispetto agli eventi di tassonomia di Tabella 1.

Passaggio

Giocatore che calcia la palla e consente ad un compagno di squadra di ottenere il possesso palla. La regola formale è data da una sequenza di calcio palla e possesso palla, di giocatori entrambi della stessa squadra.

$$
\begin{aligned}
&< ID, Pass, < PossessingPlayer, p_i, t >, < ReceivingPlayer, p_j, t+k >, < PossessedObject, b, t >> \\
&player(p_i), player(p_j), ball(b), Distance(p_i, b, t) < Th \\
&team(p_i) = team(p_j) \\
&< ID, Pass, T =< ID, KickingTheBall, < KickingPlayer, p_i, t >, < KickedObject, b, t >> \\
&THEN \\
&< ID, BallPossession, < PossessingPlayer, p_j, t+k >, < PossessedObject, b, t >>> \\
&k < Th3 \text{ (durata arbitraria passaggio, soglia a scelta)}
\end{aligned}
$$

Passaggio filtrante

Giocatore che calcia la palla e consente ad un compagno di squadra di ottenere il possesso palla riuscendo a superare l'ultimo difensore della squadra avversaria. La regola formale è simile a quella del Pass ma con in più la clausola che il giocatore ricevente la palla si trovi più vicino alla porta avversaria in un istante di tempo successivo a quello di inizio del passaggio, ma all'interno di un intervallo delta massimo.

$$
\begin{aligned}
&< ID, FilteringPass, T = \\
&< ID, Pass, < PossessingPlayer, p_i, t >, < ReceivingPlayer, p_j, t+k >, < PossessedObject, b, t >>> \\
&player(p_i), player(p_j), ball(b), team(p_i) = team(p_j), \\
&\forall k, player(p_k), team(p_k) \neq team(p_j), goal(g, p_k), D(p_j, g, t+k) < D(p_k, g, t+k)
\end{aligned}
$$

Cross

Giocatore che calcia la palla da una zona laterale del terreno di gioco e spinge la palla nella zona di interno campo. La regola formale è simile a quella del pass con in più clausole spaziali che costringono l'inizio del passaggio ad avvenire nelle fasce laterali, mentre la fine del passaggio nella regione dell'interno campo.

$$
\begin{aligned}
&< ID, Cross, T = \\
&< ID, Pass, < PossessingPlayer, p_i, t >, < ReceivingPlayer, p_j, t+k >, < PossessedObject, b, t >>> \\
&player(p_i), player(p_j), ball(b), team(p_i) = team(p_j), \\
&y1_{min} < (p_i)_y(t) < y_{max} \\
&(p_i)_x(t) < x1_{min} V (p_i)_x(t) > x1_{max} \\
&y2_{min} < (p_j)_y(t+k) < y_{max} \\
&x2_{min} < (p_j)_x(t+k) < x2_{max}
\end{aligned}
$$

Tackle

Tentativo avvenuto con successo di togliere il controllo palla ad un giocatore della squadra avversaria. La regola formale è data da un tackle atomico seguito dall'evento possesso palla.

$$
\begin{aligned}
&< ID, WonTackle, < PossessingPlayer, p_i, t >, < TacklingPlayer, p_j, t >< PossessedObject, b, t >> \\
&player(p_i), player(p_j), ball(b), \\
&team(p_i) \neq team(p_j) \\
&< ID, WonTackle, T = \\
&< ID, Tackle, < PossessingPlayer, p_i, t >, < TacklingPlayer, p_j, t >< PossessedObject, b, t >> \\
&THEN \\
&< ID, BallPossession, < PossessingPlayer, p_j, t+k >, < PossessedObject, b, t+k >> \\
&k < Th3
\end{aligned}
$$

Tiro fuori

Giocatore che calcia la palla in direzione porta ma la palla va fuori, ai lati della porta o in altro, sopra la traversa. La regola formale è data da un tiro seguito da un evento di palla fuori.

$$
\begin{aligned}
&< ID, Shot, < ShootingPlayer, p_i, t >, < PossessedObject, b, t+k >> \\
&player(p_i), ball(b), \\
&Distance(p_i, b, t) < Th \\
&< ID, Shoot, T =< ID, KickingTheBall, < KickingPlayer, p_i, t >, < KickedObject, b, t >> \\
&THEN \\
&< ID, BallOut, < Ball, b, t+k >>> \\
&k < Th3
\end{aligned}
$$

Tiro goal

Giocatore che calcia la palla in direzione porta e riesce a fare goal. La regola formale è data da un tiro seguito da un goal.

$$
\begin{aligned}
&< ID, ShotAndGoal, < ScoringPlayer, p_i, t >, < PossessedObject, b, t + k >> \\
&player(p_i), ball(b), \\
&< ID, ShotAndGoal, T = \\
&< ID, KickingTheBall, < KickingPlayer, p_i, t >, < KickedObject, b, t >> \\
&THEN \\
&< ID, Goal, < Ball, b, t + k >>> \\
&k < Th3
\end{aligned}
$$

Tiro parato

Giocatore che calcia la palla in direzione porta ma il portiere ne blocca il tentativo di rete. La regola formale è data da un calcio palla seguito da una deflessione palla.

$$
\begin{aligned}
&< ID, SavedShot, < KickingPlayer, p_i, t >, < GoalKeeper, p_j, t + k >, < KickedObject, b, t >> \\
&player(p_i), ball(b), GoalKeeper(p_j), \\
&< ID, SavedShot, T =< ID, KickingTheBall, < KickingPlayer, p_i, t >, < KickedObject, b, t >> \\
&THEN \\
&< ID, BallDeflectionByPlayer, p_j, t + k > \\
&t_k - t_i < t_h
\end{aligned}
$$

Passaggio sfociato in goal

Giocatore che riceve la palla per Giocatore che calcia la palla in direzione porta ma la palla va fuori, ai lati della porta o in altro, sopra la traversa. La regola formale è data da un tiro seguito da un evento di palla fuori.mezzo di un calcio palla e che, dopo aver effettuato un controllo palla, calcia in porta e fa goal. La regola formale è un passaggio seguito da goal.

$$
\begin{aligned}
&< ID, PassThenGoal, \\
&< AssistingPlayer, p_i, t >, < ScoringPlayer, p_j, t + h_2 >, < PossessedObject, b, t + k >> \\
&player(p_i), player(p_j), ball(b), \\
&Distance(p_i, b, t) < Th \\
&team(p_i) = team(p_j) \\
&< ID, PassThenGoal, T =< ID, Pass, \\
&< PossessingPlayer, p_i, t >, < ReceivingPlayer, p_j, t + h_1 >, < PossessedObject, b, t >> \\
&THEN \\
&< ID, ShootOnGoal, < ScoringPlayer, p_j, t + h_2 >, < PossessedObject, b, t + k >>> \\
&k > h_2 > h_1, k < Th3
\end{aligned}
$$

Filtrante sfociato in goal

Giocatore che riceve la palla oltre la linea dell'ultimo difensore della squadra avversaria, per mezzo di un calcio palla, e dopo aver effettuato un controllo palla, calcia in porta e fa goal. La regola formale è data da passaggio filtrante seguito da goal.

$$
\begin{aligned}
&< ID, FilteringPassThenGoal, \\
&< AssistingPlayer, pi, t >, < ScoringPlayer, pj, t + h_2 >, < PossessedObject, b, t + k >> \\
&player(p_i), player(p_j), ball(b), \\
&Distance(p_i, b, t) < Th \\
&team(p_i) = team(p_j) \\
&< ID, FilterPassThenGoal, T =< ID, FilteringPass, \\
&< PossessingPlayer, p_i, t >, < ReceivingPlayer, p_j, t + h_1 >, < PossessedObject, b, t >> \\
&THEN \\
&< ID, ShootOnGoal, < ScoringPlayer, p_j, t + h_2 >, < PossessedObject, b, t + k >>> \\
&k > h_2 > h_1, k < Th3
\end{aligned}
$$

Cross sfociato in goal

Giocatore che calcia la palla da una zona laterale del terreno di gioco e spinge la palla nella zona di interno campo, consentendo ad un compagno di squadra di ottenerne Il possesso, di tirare in porta e fare goal. L'evento formale è dato dall'evento cross seguito da goal.

$$
\begin{aligned}
&< ID, CrossThenGoal, \\
&< AssistingPlayer, p_i, t >, < ScoringPlayer, p_j, t + h_2 >, < PossessedObject, b, t + k >> \\
&player(p_i), player(p_j), ball(b), \\
&Distance(p_i, b, t) < Th \\
&team(p_i) = team(p_j) \\
&< ID, CrossThenGoal, T =< ID, Cross, \\
&< PossessingPlayer, p_i, t >, < ReceivingPlayer, p_j, t + h_1 >, < PossessedObject, b, t >> \\
&THEN \\
&< ID, ShootOnGoal, < ScoringPlayer, p_j, t + h_2 >, < PossessedObject, b, t + k >>> \\
&k > h_2 > h_1, k < Th3
\end{aligned}
$$

Etalis e logiche temporali

Una grossa sfida per l'identificazione e l'emissione degli eventi complessi è stata quella di trovare un modo per fare quello che in letteratura è noto come *Complex Event Processing* (CEP). Prima di tutto, è stato necessario capire se bastava semplicemente dare un ordinamento agli eventi (in fondo, per emettere un evento Pass mi basta sapere che prima è occorso KickingTheBall, poi in successione BallPossession) oppure se era necessario avere una descrizione quantitativa in maniera da definire quale evento è occorso prima di altri e, soprattutto, di quanto. Sistemi di logiche temporali, descritti al capitolo 3, si assestano bene per questo fine, in quanto riescono a descrivere la dinamica con una conoscenza ontologica descrittiva. Per risolvere il problema sul trattamento degli eventi è possibile adottare due strategie: logiche temporali qualitative, ed essere in grado di risolvere relazioni basate su un ordinamento temporale, oppure usare una metrica del tempo per esprimere clausole temporali. Le clausole possono, quindi, essere categorizzate in due:

- Clausole di tipo qualitativo o a ordinamento degli eventi
- Clausole di tipo quantitativo

In letteratura sono presenti numerose scelte possibili ma, molte di esse sono rimaste concettualizzate, e solo poche sono state utilizzate su problemi del mondo reale, e sviluppate per essere eseguite e sottoposte a testing sotto forma di debug. Quelle che si sono avvicinate al caso di studio sono ITL (Interval Temporal Logic) e TILCO. ITL presenta un unità di tempo che è un intervallo, il che risulta utilizzabile nel presente dominio di studio, ma l'intervallo ha una lunghezza che è pari al numero degli stati nella sequenza degli eventi, e non ha una metrica per definire clausole temporali in maniera quantitativa. TILCO, invece, riesce a mappare un numero intero ad ogni

istante temporale, e anch'esso ha come unità fondamentale un intervallo di tempo. Ma a differenza di ITL, per la finestra temporale è specificabile il tempo di inizio e di fine. Inoltre presenta alcune funzionalità comode come l'introduzione di nuovi operatori logici. Implementativamente sono stati valutati AnaTempura, come implementazione di ITL, ma non soddisfa i criteri perché è valido solo per quei problemi in cui basta rappresentare il passaggio del tempo come passaggio di stati, mentre nel caso in esame è altamente necessario poter specificare anche finestre temporali di validità all'interno del quale un evento può essere emesso oppure no. E' infatti utile definire queste finestre temporali entro cui in evento può avvenire, parametrizzandole mediante soglie. Avere una soglia consente di poter evitare l'emissione di eventi complessi che combinino eventi atomici molto disanti tra di loro. Un evento di passaggio semplice che, come vedremo, è formalizzato come calcio palla, seguito da possesso palla, può avvenire all'interno di una finestra di 500 frame e il sistema di logiche temporali ha bisogno di una metrica quantitativa per controllare questa clausola.

Etalis, invece, implementazione di TILCO, risponde bene alle specifiche richieste.

Tool	Logic System	Metric for time	New Predicates	Compositional operators	Doc	Environments
AnaTempura	ITL	N.A.	Limited	No	Poor	Linux, Windows
Etalis	~TILCO	✓	Yes	Yes	Good	Linux, Windows

Tabella 5 - La tabella mostra la comparazione tra i due sistemi di processamento di eventi AnaTempura ed Etalis per il problema di logica temporale da affrontare.

Dopo questa fase di valutazione si è pertanto deciso di usare sistemi a logiche temporali con clausole di tipo quantitativo, che avessero una metrica di valutazione delle finestre temporali flessibile, e implementativamente Etalis, estensione di Prolog.

L'architettura di alto livello di questo sistema per eventi complessi è composta da alcuni sottomoduli software specificamente demandati a svolgere un'operazione singola: preprocessamento dei dati per ottenere il formato proprietario di Etalis, l'operazione di riconoscimento vera e propria, e il il postprocessamento per ottenere un formato dei dati coerente con il sistema di validazione.

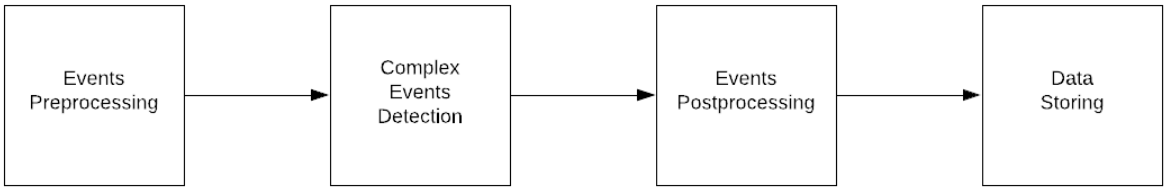


Figura 16 - Architettura di alto livello del sistema per eventi complessi.

Il sistema dei complessi si può pensare come un unico strato applicativo che non interagisce mai con lo strato dei dati: esso non contatterà mai alcuna base dati ma si occuperà di trattare e trasformare la lista di eventi atomici in ingresso, che ha natura transiente e non persistente. Nello specifico il linguaggio e le tecnologie coinvolte nel processo sono le seguenti:

- ETALIS, come estensione di PROLOG
- SWIPL, come ambiente di sviluppo per PROLOG
- Pyswip, come interfaccia Python a PROLOG
- Python, per il processamento base dei dati

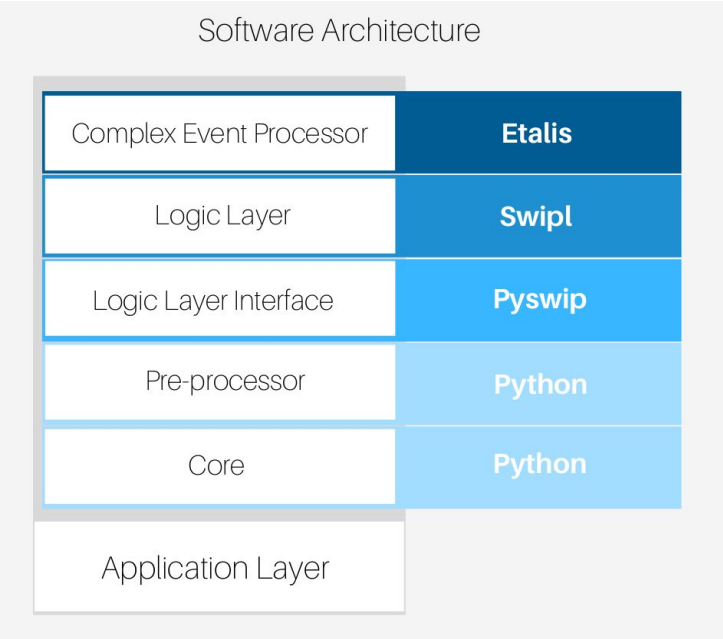


Figura 17 - Architettura di livello software per il sistema di riconoscimento per eventi complessi.

Etalis e suoi operatori logici

Etalis, come specificato nei capitoli precedenti, si è dimostrato essere il fulcro per il processamento degli eventi semantici di tipo atomico, e per la rilevazione di quelli di tipo complesso. Infatti, poiché strumento specifico per questo compito, ha permesso di raggiungere gli obiettivi preposti con dei risultati che andremo ad esaminare nei capitoli successivi, ma tenendo in considerazione alcuni fattori chiave e di qualità nello sviluppo di sistemi software quali per esempio prestazioni, consumo di risorse, nonché comprensione, leggibilità e portabilità del codice sorgente.

Struttura di una regola di Etalis

Una regola di Etalis ha una struttura definita in questo modo:

```
EventoOutput <-
EventoInput1 <LOGIC_OP> EventoInput2  [<LOGIC_OP> EventoInputN]
WHERE (ListaDiClausoleLogiche)
```

Dove:

- EventoInput, è l'evento in ingresso
- EventoOutput, è l'evento in uscita
- <LOGIC_OP> è l'operatore logico che può essere per esempio seq, not, e così via
- WHERE, introduce la lista di clausole
- ListaDiClausoleLogiche, contiene le clausole

Architettura di Etalis

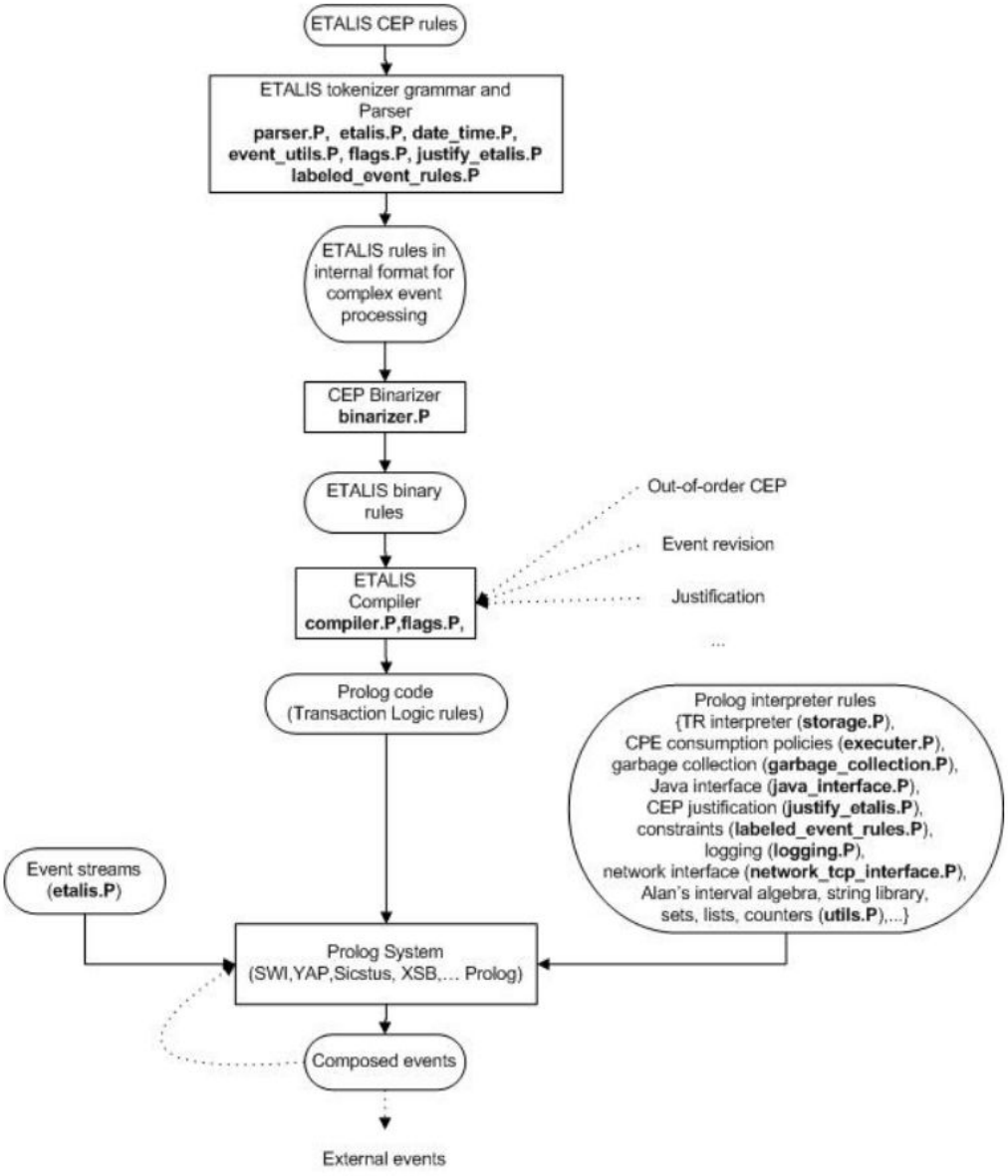


Figura 18 - Architettura di Etalis, e parti che garantiscono il suo *Complex Event Processing*.

Capitolo 5

Implementazione

5.1 Framework di riconoscimento eventi

Per l'implementazione del sistema, come specificato nei capitoli precedenti, sono state usate due metodologie diverse ma coerenti e complementari. Per la parte degli eventi atomici si è fatto uso di un sistema esperto basato su regole, giustificato dal fatto che in questo contesto è possibile utilizzare una conoscenza esperta di dominio. Per la parte degli eventi complessi si è fatto uso di un sistema di ragionamento logico, anch'esso implementato con regole. Mentre la parte di implementazione del sistema per atomici è stata sviluppata in un altro lavoro di tesi parallelo, quella che si andrà qui a descrivere è quella del sistema per eventi complessi.

La sua fase di implementazione è composta da tre parti: preprocessamento, sviluppo del sistema basato su regole mediante Etalis, ed infine una fase di postprocessamento. Entrambe le fasi che agiscono ai margini del sistema di regole sono adibite ad eseguire l'operazione che in letteratura è nota come ETL, ossia la composizione in sequenza di estrazione, trasformazione e caricamento: un processo utile a trasformare i dati in un maniera più utile e funzionale al suo utilizzo.

Preprocessamento

Il sistema di riconoscimento per eventi complessi prende in ingresso un documento in formato dati XML contenente la lista degli eventi atomici che sono stati rilevati dal sistema di riconoscimento per eventi atomici. Dal momento che il cuore del sistema per eventi complessi è Etalis ed esso prende in ingresso uno stream di eventi, in un formato proprietario, è necessario eseguire un'operazione di preprocessamento per garantire piena compatibilità.

Il modulo software che si occupa di eseguire questa operazione di preprocessamento è uno script scritto con il linguaggio di programmazione Python, alla quale è stata abbinata una API per la parsificazione del contenuto del documento in formato dati XML. Il nome di questa interfaccia applicativa è *ElementTree API*.

Durante la fase di progettazione, rispetto a questo compito, sono state valutate altre opzioni come per esempio quella di integrare direttamente nel sistema di

riconoscimento degli eventi atomici il modulo necessario ad esportare la lista degli eventi secondo il formalismo richiesto da Etalis. Il vantaggio di questa soluzione sarebbe stato quello di creare un secondo modulo da aggiungere in parallelo a quello già presente, il cui obiettivo era quindi solo quello di trasformare in due formati dati persistenti quell'informazione presente in memoria. Mentre, la soluzione poi sviluppata, ha richiesto di parsificare il contenuto del documento XML, caricare in memoria i dati e, solo a quel punto, esportare gli stessi nel formato dati proprietario di Etalis.

Tuttavia, si è preferita la seconda scelta anche se, a livello di prestazioni, peggiore, per un'altra ragione ben più profonda e radicata. Il sistema di riconoscimento per eventi complessi vuole essere quanto più possibile distaccato ed indipendente dal sistema di riconoscimento per eventi atomici. Inoltre, il sistema degli atomici in questa maniera ha il vantaggio di mettere in uscita un formato dati human readable.

Ambiente di lavoro di Etalis

Nel seguente, per ambiente di lavoro, si intende l'insieme di documenti e di direttori, utili all'organizzazione del modulo software che si vuole sviluppare. I documenti si riferiscono al codice sorgente del suddetto modulo.

L'ambiente di lavoro di Etalis, essendo esso, perlopiù, un sistema di nicchia e poco supportato dalla comunità degli sviluppatori, comprende pochi elementi ma essenziali, che comunque permettono di suddividere logicamente il processamento che si andrà a fare:

- Un documento per caricare l'estensione Etalis rispetto a Prolog.
- Un documento per configurare alcuni parametri di inizializzazione.
- Tre documenti per il sistema di regole.
- Un documento con lo stream di eventi.
- Un log per fornire allo sviluppatore indicazioni sull'andamento globale del processamento.

5.2 Il sistema di regole

Qui di seguito andrò a descrivere il sistema di regole implementato in Etalis. Esso è composto da un totale di 64 regole. L'ordine con cui vengono processate le regole è importante perché stabilisce la priorità nell'emissione degli eventi. Ci sono, inoltre, tre elementi chiave da premettere che svolgono ruolo chiave nell'implementazione delle regole e per l'ottenimento del risultati desiderati: essi sono l'utilizzo della garbage collection, l'introduzione di eventi subordinati, e l'introduzione di eventi di servizio.

Garbage Collection

La Garbage Collection è resa disponibile dall'estensione ETALIS e consente di eliminare dalla memoria tutti eventi che sono arrivati dal flusso in ingresso. Essa è utile perché consente da una parte di mantenere in memoria un numero limitato di eventi, dall'altro di azzerare gli eventi in ingresso e arrestare eventuali emissioni, in casi non desiderati.

Qui di seguito un caso eclatante che ne mostra l'efficacia del suo utilizzo:

1	KickingTheBall
2	Tackle
3	[...]
4	KickingTheBall
5	BallPossession
6	BallPossession
7	BallPossession

Gli eventi 4 e 5 vengono accoppiati per fare Pass, essi vengono consumati e non possono più intercorrere per l'emissione di un evento che ne faccia il loro utilizzo. All'occorrere dell'evento 6 vengono processate tutte le regole e ancora una volta l'evento KickingTheBall, il numero 1, unito a BallPossession genererà un Pass.
In realtà non vi è un passaggio tra il giocatore che esegue il calcio palla del KickingTheBall 1 e chi fa controllo palla dell'evento 6, quindi è necessaria una forma di riazzeramento che impedisca l'emissione di questo evento, così come di altri indesiderati. La garbage collection viene attivata, laddove specificata, poco prima dell'emissione di un evento, riazzerando la storia passata. Nel caso in esame è specificata come ultima clausola per l'emissione dell'evento Pass. Al momento di

attivazione della garbage collection per l'evento Pass tutte le clausole per l'emissione di questo evento sono state verificate ed è dunque sicuro che questo evento debba essere emesso. Prima di essere emesso però viene riazzerata la storia passata e la lista di eventi nello stream di Etalis diventerà:

1	Pass
2	BallPossession
3	BallPossession

Eventi subordinati

Un evento subordinato è un evento che viene emesso così come tutti gli altri da ETALIS ma che, a differenza degli altri, non viene riportato in uscita e che viene utilizzato come stato intermedio tra gli eventi atomici di ingresso e gli eventi complessi in uscita. La lista di eventi subordinati è rappresentata nella tabella successiva.

Evento subordinato	Spiegazione
FilteringPass_and_Pass	Evento aggregato di passaggio filtrante e passaggio semplice.
Cross_and_Pass	Evento aggregato di cross e passaggio semplice.
FilteringPass_and_Cross_and_Pass	Evento aggregato di passaggio filtrante, cross e passaggio semplice.
TackleSC	Evento di raggruppamento di tackle atomici.

Tabella 6 - Eventi subordinati

Eventi di servizio

Un evento di servizio non è propriamente un evento, ma è pressoché utile per eseguire conversioni di valori o come dizionario di stringhe.

Evento di servizio	Spiegazione
getLeft	Restituisce la stringa "Left"
getRight	Restituisce la stringa "Right"
getSHframe	Restituisce il frame di inizio del secondo tempo

latLngRegion	Restituisce la regione del terreno di gioco
getPlayerPosSide	Restituisce l'area del terreno di gioco in cui si trova un giocatore
getTeamSide	Restituisce il lato di gioco di una squadra
getGK	Restituisce l'id dei portieri
getTrue	Restituisce la stringa "True"
getFalse	Restituisce la stringa "False"
getInsideGoalArea	Restituisce la stringa "InsideGoalArea"
getInsidePenaltyArea	Restituisce la stringa "InsidePenalityArea"
getFromOutsideArea	Restituisce la stringa "FromOutsideArea"
getOnFoul	Restituisce la stringa "OnFoul"
getOnPenality	Restituisce la stringa "OnPenalty"

Tabella 7 - Eventi di servizio

L'evento complesso ShotOut

L'evento complesso *tiro fuori* indica un tiro che non è direzionato nello specchio della porta e che quindi è finito fuori, plausibilmente o sopra o ai lati della porta. E' formalizzato come sequenza di *calcio palla* (KickingTheBall) e *tiro fuori* (BallOut), e tra le clausole viene verificato che, al momento in cui avviene il tiro, il giocatore che calcia la palla si trovi nella metà campo avversaria, ossia sta effettuando un tiro verso la porta avversaria e non nella propria porta.

Rule
shotOut(Kf,BOf,KpId,KtId) <- kickingTheBall(Kf,KpId,KtId,Kx,Ky) seq ballOut(BOf) where (BOf>Kf,getPlayerPosSide(Kx,Pps), getSHframe(SHframe),getTeamSide(KtId,Kf,SHframe,TeamSide), TeamSide\=Pps).
Arguments

Kf = KickingTheBall frame
KpId = KickingTheBall player Id
KtId = KickingTheBall team Id
Kx = KickingTheBall position over the x axis
Ky = KickingTheBall position over the y axis
BOf = BallOut frame

sL'evento complesso Shot

L'evento *tiro* (Shot) è un evento che esprime l'intenzione di un calciatore di tirare verso lo specchio della porta per il fine di fare goal. La sua emissione è possibile in tre casi distinti:

- In concomitanza con l'emissione di ShotOut
- In concomitanza con l'emissione di ShotThenGoal
- In concomitanza con l'emissione di SavedShot

Qui di seguito l'esempio della regola che permette l'emissione di Shot quando in concomitanza con l'evento *tiro fuori* (ShotOut).

Rule

```
shot(Kf,BOf,KpId,KtId) <-
kickingTheBall(Kf,KpId,KtId,Kx,Ky)
seq
ballOut(BOf)
where (BOf>Kf,getPlayerPosSide(Kx,Pps),
getSHframe(SHframe),getTeamSide(KtId,Kf,SHframe,TeamSide),
TeamSide\=Pps).
```

Arguments

Kf = KickingTheBall frame
KpId = KickingTheBall player Id
KtId = KickingTheBall team Id
Kx = KickingTheBall position over the x axis
Ky = KickingTheBall position over the y axis
BOf = BallOut frame

L’evento complesso ShotThenGoal

L’evento *tiro poi goal* esprime un tiro che è sfociato in goal, ossia un tiro che è finito all’interno della regione della porta, ed è ottenuto come sequenza di *calcio palla* (KickingTheBall) e *palla in rete* (Goal).

Rule
shotThenGoal(Kf,Gf,KpId,KtId) <- kickingTheBall(Kf,KpId,KtId,Kx,Ky) seq goal(Gf,GsId,GtId) where (Gf>Kf,Kf-Gf<500).
Arguments
Kf = KickingTheBall frame Bf = BallPossession frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id BpId = BallPossession player Id Gf = Goal frame GsId = Goal scorer Id GtId = Goal scorer team Id

L’evento complesso FilteringPass

L’evento *passaggio filtrante* è un evento che è ottenuto come sequenza di *calcio palla* (KickingTheBall) e *possesso palla* (BallPossession) con le stesse clausole presenti per l’evento passaggio, ma in più la condizione che chi riceve il passaggio si trova oltre la linea difensiva, costruita sulla base della posizione dell’ultimo giocatore, a eccezione del portiere, della squadra avversaria. Inoltre, poiché i giocatori cambiano lato del campo di gioco nel passaggio tra la prima metà tempo e la seconda, è necessario aggiungere delle clausole aggiuntive che verificano l’attuale posizione della squadra nel campo di gioco, ma più nello specifico a controllare in maniera coerente che il giocatore che riceve la palla sia oltre la linea dell’ultimo difensore della squadra opponente nel lato corretto del campo. Per brevità è qui riportata la regola valida per la squadra che è posizionata a riposo nel lato sinistro del campo, ma ne esiste una controparte simmetrica implementata.

Rule

filteringPass(Kf,Bf,KpId,KtId,BpId) <-
kickingTheBall(Kf,KpId,KtId,Kx,Ky)
seq
ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdptId,Bootdpx,Bootdpy)
where KtId=BtId,Bf>Kf,Bf-Kf<500,
getLeft(Left),getSHframe(SHframe),getTeamSide(BtId,Bf,SHframe,TeamSide),
TeamSide=Left,Bootdpx<Bx).

Arguments

Kf = KickingTheBall frame
KpId = KickingTheBall player Id
KtId = KickingTheBall team Id
Kx = KickingTheBall position over the x axis
Ky = KickingTheBall position over the y axis
Bf = BallPossession frame
BpId = BallPossession player Id
BtId = BallPossession team Id
Bx = BallPossession position over the x axis
By = BallPossession position over the y axis
BootdpId = BallPossession other team defensive player Id
Bootdpx = BallPossession other team defensive player position over the x axis
Bootdpy = BallPossession other team defensive player position over the y axis

Note

Per la definizione di questo evento si è usata una soglia di 500 frames, un intervallo temporale che occorre tra il calcio alla palla e il possesso palla. In questo caso il valore è giustificabile dalla dinamica dell'evento. Esso è un passaggio che normalmente copre lunghe distanze e pertanto richiede qualche secondo in più per poter riuscire e, inoltre, è un passaggio che implica il superamento dell'ultimo difensore della squadra avversaria. I parametri Bootdpx e Bootdpy rappresentano propria le coordinate (x, y) che rappresentano la posizione dell'ultimo difensore della squadra avversaria.

L’evento complesso Cross

L’evento complesso cross è ottenuto come combinazione sequenziale di calcio-palla (KickingTheBall) e possesso-palla (BallPossession) con la clausola che il calcio-palla sia avvenuto in una regione del campo da calcio chiamata *fascia* e che la ricezione della palla e quindi la posizione del possesso-palla sia ricaduta all’interno di un’altra regione del campo detta *center*.

Rule
cross(Kf,Bf,KpId,KtId,BpId,success) <- kickingTheBall(Kf,KpId,KtId,Kx,Ky) seq ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdptId,Bootdpx,Bootdpy) where (KtId=BtId,Bf>Kf,Bf-Kf<150,latLngToRegion(Kx,Ky,Kr),latLngToRegion(Bx,By,Br),getSide(Side),getCenter(Center),Kr=Side,Br=Center,getSuccess(success)).
Arguments
Kf = KickingTheBall frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id Kx = KickingTheBall position over the x axis Ky = KickingTheBall position over the y axis Bf = BallPossession frame BpId = BallPossession player Id BtId = BallPossession team Id Bx = BallPossession position over the x axis By = BallPossession position over the y axis BootdpId = BallPossession other team defensive player Id Bootdpx = BallPossession other team defensive player position over the x axis Bootdpy = BallPossession other team defensive player position over the y axis
Note
Le funzioni di supporto latLngToRegion, getSide, getCenter, getSuccess consentono rispettivamente di mappare le coordinate di un giocatore o della palla con una regione del campo di gioco, e poi di ottenere le stringhe Side, Center, e Success per la corretta esportazione dell’evento.

Evento FilteringPass e Cross e Pass

Questo è il primo degli eventi subordinati descritti precedentemente. Esso descrive un evento che è sia *passaggio filtrante* (FilteringPass), sia *cross* (Cross), sia *passaggio* (Pass). Ossia possiamo considerare questo come un evento aggregato, pensato con l'intento di semplificare un suo eventuale accostamento con un cronologicamente successivo evento di *palla in rete* (Goal). L'emissione di questo evento è possibile con dodici regole: esprimono un passaggio che va da fascia laterale a interno campo o area di rigore o area di porta, e poi definiti specularmente per le due squadre, e con outcome riuscito o fallito. La mappatura completa è descritta in maniera esclusiva dai casi:

- Passaggio filtrante riuscito con cross da fascia laterale a interno campo, effettuato da un giocatore della squadra di casa

- Passaggio filtrante non riuscito con cross da fascia laterale a interno campo, effettuato da un giocatore della squadra di casa

- Passaggio filtrante riuscito con cross da fascia laterale ad area di rigore effettuato da un giocatore della squadra di casa

- Passaggio filtrante non riuscito con cross da fascia laterale ad area di rigore effettuato da un giocatore della squadra di casa

- Passaggio filtrante riuscito con cross da fascia laterale ad area di porta, effettuato da un giocatore della squadra di casa

- Passaggio filtrante non riuscito con cross da fascia laterale ad area di porta, effettuato da un giocatore della squadra di casa

- Passaggio filtrante riuscito con cross da fascia laterale a interno campo, effettuato da un giocatore della squadra in trasferta

- Passaggio filtrante non riuscito con cross da fascia laterale a interno campo, effettuato da un giocatore della squadra in trasferta

- Passaggio filtrante riuscito con cross da fascia laterale ad area di rigore effettuato da un giocatore della squadra in trasferta

- Passaggio filtrante non riuscito con cross da fascia laterale ad area di rigore effettuato da un giocatore della squadra in trasferta

- Passaggio filtrante riuscito con cross da fascia laterale ad area di porta, effettuato da un giocatore della squadra in trasferta

- Passaggio filtrante non riuscito con cross da fascia laterale ad area di porta, effettuato da un giocatore della squadra in trasferta

Qui riportata solo una delle 12 regole per ragioni di compattezza.

Regola
filteringPass_and_cross_and_pass(Kf,Bf,KpId,KtId,BpId,Co) <- kickingTheBall(Kf,KpId,KtId,Kx,Ky) seq ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdptId,Bootdpx,Bootdpy) where (KtId=BtId,BpId\=KpId,Bf>Kf,Bf-Kf<500,getLeft(Left),getSHframe(SHframe),getTeamSide(BtId,Bf,SHframe,TeamSide),TeamSide=Left,Bootdpx<Bx,latLngToRegion(Kx,Ky,Kr),latLngToRegion(Bx,By,Br),getSide(Side),getCenter(Center),Kr=Side,Br=Center,getTrue(Co),call_gc).
Argomenti
Kf = KickingTheBall frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id Kx = KickingTheBall position over the x axis Ky = KickingTheBall position over the y axis Bf = BallPossession frame BpId = BallPossession player Id BtId = BallPossession team Id Bx = BallPossession position over the x axis By = BallPossession position over the y axis BootdpId = BallPossession other team defensive player Id Bootdpx = BallPossession other team defensive player position over the x axis Bootdpy = BallPossession other team defensive player position over the y axis
Eventi di servizio
getLeft(Left), restituisce la stringa “Left”

getSHframe(SHframe), restituisce il frame di inizio del secondo tempo
getTeamSide(BtId,Bf,SHframe,TeamSide), dati l'id di un giocatore, la sua squadra, e il frame di inizio del secondo tempo, restituisce il lato della sua metà campo
latLngToRegion(Kx,Ky,Kr), dati le posizioni x e y di un giocatore, restituisce una stringa identificativa della regione in cui si trova il giocatore.
getSide(Side), restituisce la stringa "Side".
getCenter(Center), restituisce la stringa Center
call_gc, richiama la garbage collection

Evento FilteringPass e Pass

Questo è un evento subordinato che rappresenta un evento che è sia *passaggio filtrante* (FilteringPass) e sia *passaggio* (Pass). L'emissione di questo evento è possibile mediante due regole:

- Passaggio filtrante effettuato da un giocatore della squadra di casa
- Passaggio filtrante effettuato da un giocatore della squadra in trasferta

Regola

```
filteringPass_and_pass(Kf,Bf,KpId,KtId,BpId) <-
kickingTheBall(Kf,KpId,KtId,Kx,Ky)
Seq
ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdptId,Bootdpx,Bootdpy)
where
(KtId=BtId,BpId\=KpId,Bf>Kf,Bf-Kf<500,getLeft(Left),getSHframe(SHframe),getTeam
Side(BtId,Bf,SHframe,TeamSide),TeamSide=Left,Bootdpx<Bx, call_gc).
```

Argomenti

Kf = KickingTheBall frame
KpId = KickingTheBall player Id
KtId = KickingTheBall team Id
Kx = KickingTheBall position over the x axis
Ky = KickingTheBall position over the y axis
Bf = BallPossession frame
BpId = BallPossession player Id
BtId = BallPossession team Id
Bx = BallPossession position over the x axis
By = BallPossession position over the y axis
BootdpId = BallPossession other team defensive player Id

Bootdpx = BallPossession other team defensive player position over the x axis Bootdpy = BallPossession other team defensive player position over the y axis
Eventi di servizio
getLeft(Left), restituisce la stringa "Left", utile per verificare la posizione della squadra nel terreno di gioco getSHframe(SHframe), restituisce il frame di inizio del secondo tempo getTeamSide(BtId,Bf,SHframe,TeamSide), dati l'id di un giocatore, la sua squadra, e il frame di inizio del secondo tempo, restituisce il lato della sua metà campo call_gc, richiama la garbage collection

Evento Cross e Pass

Questo è un evento subordinato che rappresenta un evento che è sia *cross* (Cross) e sia *passaggio* (Pass). L'emissione di questo evento è possibile mediante sei regole:

- Cross riuscito da fascia laterale a interno campo
- Cross non riuscito da fascia laterale a interno campo
- Cross riuscito da fascia laterale ad area di rigore
- Cross non riuscito da fascia laterale ad area di rigore
- Cross riuscito da fascia laterale ad area di porta
- Cross non riuscito da fascia laterale ad area di porta

Regola
filteringPass_and_pass(Kf,Bf,KpId,KtId,BpId) <- kickingTheBall(Kf,KpId,KtId,Kx,Ky) Seq ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdptId,Bootdpx,Bootdpy) where (KtId=BtId,BpId\=KpId,Bf>Kf,Bf-Kf<500,getLeft(Left),getSHframe(SHframe),getTeamSide(BtId,Bf,SHframe,TeamSide),TeamSide=Left,Bootdpx<Bx, call_gc).
Argomenti
Kf = KickingTheBall frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id Kx = KickingTheBall position over the x axis Ky = KickingTheBall position over the y axis

Bf = BallPossession frame
BpId = BallPossession player Id
BtId = BallPossession team Id
Bx = BallPossession position over the x axis
By = BallPossession position over the y axis
BootdpId = BallPossession other team defensive player Id
Bootdpx = BallPossession other team defensive player position over the x axis
Bootdpy = BallPossession other team defensive player position over the y axis

Eventi di servizio

getLeft(Left), restituisce la stringa "Left"
getSHframe(SHframe), restituisce il frame di inizio del secondo tempo
getTeamSide(BtId,Bf,SHframe,TeamSide), dati l'id di un giocatore, la sua squadra, e il frame di inizio del secondo tempo, restituisce il lato della sua metà campo
call_gc, richiama la garbage collection

Evento complesso Pass

L'evento passaggio è creato come sequenziale di eventi di tipo atomico che sono il *calcio palla* (KickingTheBall) e il *possesso palla* (BallPossession) con la clausola che i giocatori coinvolti in queste due azioni siano appartenenti alla stessa squadra.

Rule

```
pass(Kf,Bf,KpId,KtId,BpId) <-
kickingTheBall(Kf,KpId,KtId,Kx,Ky)
seq
ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdpId,Bootdpx,Bootdpy)
where
(KtId=BtId, Bf>Kf, Bf-Kf <150).
```

Arguments

Kf = KickingTheBall frame
KpId = KickingTheBall player Id
KtId = KickingTheBall team Id

Kx = KickingTheBall position over the x axis
Ky = KickingTheBall position over the y axis

Bf = BallPossession frame
BpId = BallPossession player Id
BtId = BallPossession team Id
Bx = BallPossession position over the x axis
By = BallPossession position over the y axis
BootdpId = BallPossession other team defensive player Id
Bootdpx = BallPossession other team defensive player position over the x axis
Bootdpy = BallPossession other team defensive player position over the y axis

Nota

Per la definizione di questo evento è stata usata una soglia di 150 frames, un valore che determina l'intervallo temporale che occorre tra il calcio alla palla e il possesso palla.

L'evento complesso FilteringPassThenGoal

L'evento complesso passaggio-filtrante-poi-goal è ottenuto come sequenza degli eventi *passaggio filtrante* (FilteringPass) e *palla in rete* (Goal), con la clausola che chi riceve il passaggio è anche l'autore del goal. L'emissione di questo evento è ottenibile con tre regole:

- La regola appena descritta, e di cui sotto vi è la formalizzazione
- Il susseguirsi dell'evento subordinato FilteringPass e Cross e Pass, con l'evento atomico Goal
- Il susseguirsi dell'evento subordinato FilteringPass e Pass, con l'evento atomico Goal

Regola

```
filteringPassThenGoal(Kf,Bf,KpId,KtId,BpId) <-
filteringPass(Kf,Bf,KpId,KtId,BpId)
seq
goal(Gf,GsId,GtId)
where (Gf>Bf,GtId=KtId,GsId=BpId).
```

Arguments
Kf = KickingTheBall frame Bf = BallPossession frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id BpId = BallPossession player Id Gf = Goal frame GsId = Goal scorer Id GtId = Goal scorer team Id

L'evento complesso CrossThenGoal

L'evento cross-poi-goal è ottenuto come sequenza di *cross* (Cross) e *palla in rete* (Goal) con la clausola che chi riceve la palla mediante l'azione Cross sia anche l'autore del goal. L'emissione di questo evento è ottenuta con tre regole:

- La regola appena descritta, e di cui sotto vi è la formalizzazione
- La successione dell'evento subordiato Cross e Pass, e l'evento Goal
- La successione dell'evento subordinato FilteringPass e Cross e Pass, e l'evento Goal

Regola
crossThenGoal(Kf,Bf,KpId,KtId,BpId) <- cross(Kf,Bf,KpId,KtId,BpId) seq goal(Gf,GsId,GtId) where (Gf>Bf,GtId=KtId,GsId=BpId).
Arguments
Kf = KickingTheBall frame Bf = BallPossession frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id BpId = BallPossession player Id Gf = Goal frame GsId = Goal scorer Id GtId = Goal scorer team Id

Evento complesso PassThenGoal

L'evento passaggio-poi-goal è creato come sequenza di eventi di tipo *passaggio* (Pass) e *palla in rete* (Goal) con la clausola che il giocatore che riceve il passaggio è anche l'autore del goal. L'emissione di questo evento è ottenuta con quattro reogle:

- La regola appena descritta, e di cui sotto vi è la formalizzazione
- La successione dell'evento subordinato FilteringPass e Cross e Pass, e l'evento atomico Goal
- La successione dell'evento subordonati FilteringPass e Pass, e l'evento Goal
- La successione dell'evento subordinato Cross e Pass, e l'evento Goal

Regola
passThenGoal(Kf,Bf,KpId,KtId,BpId) <- pass(Kf,Bf,KpId,KtId,BpId) seq goal(Gf,GsId,GtId) where (Gf>Bf,GtId=BtId,GsId=BpId).
Argumenti
Kf = KickingTheBall frame Bf = BallPossession frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id BpId = BallPossession player Id Gf = Goal frame GsId = Goal scorer Id GtId = Goal scorer team Id

L’evento complesso Tackle

L’evento tackle è ottenuto come sequenza dell’evento atomico tackle (Tackle) e *possesso palla* (BallPossession), e specifica un esito a seconda che il giocatore che effettua il tackle, cosidetto *tackler*, riesca ad ottenere il posesso della palla oppure no.

Regola
tackleC(Tf,Tf,TvId,TvtId,TtId,TttId,true) <- tackle(Tf,TvId,TvtId,Tvx,Tvy,TtId,TttId,Ttx,Tty) seq ballPossession(Bf,BpId,BtId,Bx,By,BootdpId,BootdptId,Bootdpx,Bootdpy) where (Bf>Tf, BpId=TtId, getTrue(true)).
Argomenti
Tf = Tackle frame TvId = Tackle victim Id TvtId = Tackle victim team Id Tvx = Tackle victim position over the x axis Tvy = Tackle victim position over the y axis TtId = Tackle tackler Id TttId = Tackle tackler team Id Ttx = Tackle tackler position over the x axis Tty = Tackle tackler position over the y axis Bf = BallPossession frame BpId = BallPossession player Id BtId = BallPossession team Id Bx = BallPossession position over the x axis By = BallPossession position over the y axis BootdpId = BallPossession other team defensive player Id Bootdpx = BallPossession other team defensive player position over the x axis Bootdpy = BallPossession other team defensive player position over the y axis

Per l’emissione di questo regola è stato necessario introdurre un meccanismo di raggruppamento di eventi, che è stato possibile con l’introduzione dell’evento subordinato TackleSC. In totale, l’evento Tackle complesso è costituito da uno o più tackle atomici in sequenza.

L’evento complesso SavedShot

L’evento tiro-salvato è ottenuto come combinazione sequenziale di calcio-palla (KickingTheBall) e deflessione-palla (BallDeflection). L’evento complesso generato deve comunque fornire un’informazione circa la casistica di tiro-salvato ossia specificare l’area in cui è avvenuto il tiro.

Regola
savedShot(Kf,BDf,KpId,KtId,BDltpId,BDtId,BDx,BDy,insidePenaltyArea) <- kickingTheBall(Kf,KpId,KtId,Kx,Ky) seq ballDeflection(BDf,BDltpId,BDtId,BDx,BDy) where (BDf>Kf,BDf-Kf<250, getGK(BDtId,GKID), BDltpId=GKID,latLngToRegion(Kx,Ky,Kr), getInsidePenaltyArea(insidePenaltyArea),Kr=insidePenaltyArea).
Argomenti
Kf = KickingTheBall frame KpId = KickingTheBall player Id KtId = KickingTheBall team Id Kx = KickingTheBall position over the x axis Ky = KickingTheBall position over the y axis BDf = BallDeflection frame BDltpId = BallDeflection last touch player Id BDtId = BallDeflection last touch player team Id BDx = BallDeflection position over the x axis BDy = BallDeflection position over the y axis

Ordine di processamento delle regole

In Etalis le regole vengono processate secondo una certa priorità, che può essere definita in maniera del tutto personalizzata e flessibile dal programmatore. Nel contesto del riconoscimento di eventi sportivi, i casi speciali (o anche detti sottocasi) devono essere a priorità maggiore. Per maggiore comprensione si faccia riferimento al caso d’uso del passaggio filtrante (FilteringPass). Il passaggio filtrante è un sottocaso del passaggio semplice (Pass) e al momento del riconoscimento è importante che la sua regola venga processata prima di quella del passaggio semplice perché se così non fosse

l’evento FilteringPass non verrebbe emesso: mentre un passaggio filtrante è anche un passaggio, non si può dire il viceversa.

L’ordine con cui sono processate le regole è il seguente:

N.	Evento/i Emesso/i	Eventi richiesti
1	ShotOut	KickingTheBall + BallOut
2	ShotThenGoal	KickingTheBall + Goal
3	FilteringPass	KickingTheBall + BallPossession
4	Cross	KickingTheBall + BallPossession
5	FilteringPass_and_Cross_and_Pass	KickingTheBall + BallPossession
6	FilteringPass_and_Pass	KickingTheBall + BallPossession
7	Cross_and_Pass	KickingTheBall + BallPossession
8	Pass	KickingTheBall + BallPossession
9	Cross	KickingTheBall + KickingTheBall
10	Pass	KickingTheBall + KickingTheBall
11	Pass, FilteringPassThenGoal, CrossThenGoal, PassThenGoal	FilteringPass_and_Cross_and_Pass + Goal
12	FilteringPassThenGoal, PassThenGoal	FilteringPass_and_Pass + Goal
13	CrossThenGoal, PassThenGoal	Cross_and_Pass + Goal
14	FilteringPassThenGoal	FilteringPass + Goal
16	CrossThenGoal	Cross + Goal
17	PassThenGoal	Pass + Goal
18	SavedShot	KickingTheBall + BallDeflection
19	TackleSC	Tackle + Tackle
20	TackleSC	TackleSC + Tackle
21	TackleC	TackleSC + BallPossession

Etalis permette di svolgere in maniera così diretta e precisa il processamento di eventi complessi perché è progettato in maniera specifica per questo compito, ed è fondato su solide basi teoriche -- sistema di processamento degli eventi e logiche temporali -- che ne garantiscono il principio del suo funzionamento. Tuttavia, presenta, dei forti limiti nel momento in cui è necessario uscire dal suo contesto di utilizzo. Infatti, non essendo un linguaggio di programmazione di tipo procedurale, anzi non essendo del tutto un linguaggio di programmazione, risulta complicato se non impossibile, in alcuni casi, riuscire a soddisfare alcune condizioni che il sistema richiede. Nel lavoro compiuto ci sono almeno due casi importanti che sono stati affrontati e che adesso andremo a discutere, che permettono di capire quali sono i limiti che Etalis presenta, ma anche i vantaggi che esso presenta come per esempio il suo livello di astrazione, che ha permesso in un caso specifico di risolvere una problematica non indifferente.

Caso FilteringPass

Per la costruzione delle regole utili all'individuazione dell'evento complesso FilteringPass vi è una clausola delicata e molto complicata da gestire con Etalis. Prima di addentrarci direttamente nella specificazione della suddetta clausola, è bene fare una premessa che contestualizza il problema. Il passaggio-filtrante (FilteringPass) è di base un evento passaggio (Pass), quindi all'occorrenza di questo evento vi è un giocatore che calcia la palla, che chiameremo mittente, e un'altro giocatore che riceve la palla, che chiameremo destinatario. La clausola aggiuntiva che specializza il passaggio come passaggio-filtrante, che riprende la definizione ufficiale del gioco del calcio, dice che il destinatario al momento in cui riceva la palla, e quindi tocca la palla col piede, deve essere in una posizione del campo che supera la linea difensiva della squadra avversaria o, in altre parole, deve superare la linea dell'ultimo uomo della squadra avversaria, salvo il portiere. La condizione è che subito dopo il passaggio-filtrante il giocatore si trovi in un duello uno-a-uno contro il portiere della squadra avversaria.

Focalizziamoci, nuovamente, sulla regola. La formalizzazione di questo evento dice che il giocatore che riceve la palla deve trovarsi oltre la linea difensiva della squadra opponente. Dal punto di vista della programmazione e dello sviluppo software necessario a verificare questa condizione significa andare a controllare che la posizione del giocatore rispetto all'asse di maggiore estensione del campo da calcio sia superiore a quella di tutti i giocatori della squadra opponente. Questo genere di operazione, ossia il controllo in maniera iterativa su tutti i giocatori, ma anche l'estrazione in un certo istante di tempo (frame) di

tutte le posizioni della squadra opponente non è possibile con Etalis in quanto non presenta proprio nessuna caratteristica che permetta di accedere a questi dati.

La soluzione a questo problema è stata ottenuta aggiungendo all'evento BallPossession, che coincide con l'evento di terminazione del passaggio, un dato relativo alla posizione dell'ultimo giocatore della squadra avversaria, detto OutermostOtherTeamDefensivePlayer. Questo dato viene ovviamente calcolato prima, e reso disponibile nel file in input che Etalis riceve per eseguire il suo processamento. Per concludere, questo esempio dimostra quanto Etalis presenta alcuni limiti e in questo contesto di utilizzo è stato necessario preprocessare un dato.

Caso Cross

Un altro caso di notevole interesse è quello dell'evento complesso Cross. Anche questo evento, di base, come il passaggio filtrante, non è altro che un passaggio, ma la condizione particolare che lo specializza e lo differenzia è relativa alla posizione del campo in qui esso avviene. Il cross infatti deve iniziare in un'area specifica del campo da calcio che è la fascia, mentre deve terminare in un'altra area che è quella dell'interno-campo. Per riuscire a riconoscere questo evento Etalis avrebbe bisogno di riuscire a mappare, a partire dalle posizioni dei giocatori, posizioni che vengono fornite in input, le regioni del campo. Dovrebbe, in altri termini, riuscire a derivare il nome dell'area del campo da calcio in cui un dato giocatore, a compendio della sua posizione, si trovi. Con un linguaggio di programmazione procedurale basterebbe una funzione di trasformazione, che a partire dai dati della posizione, fornisce in output una stringa con il nome. In Etalis ciò non è possibile ma esiste un meccanismo che, con dovute modifiche, è in grado di fornire l'associazione discussa appena sopra. Con Etalis, è, infatti, possibile creare degli eventi che, presi in ingresso delle coordinate, forniscano in uscita un valore testuale, sintatticamente una stringa e semanticamente il nome della regione. Quello che si è andato a fare è stato definire degli eventi aggiuntivi, sempre internamente ad Etalis, non quindi forniti in input al sistema, che fossero in grado di mappare quanto richiesto, specificando delle condizioni spaziali. Questi eventi aggiuntivi sono stati aggiunti nel file di inizializzazione e non intaccano minimamente nè la logica di utilizzo, ne l'interpretazione dei risultati. Un caso simile è stato sviluppato dai creatori stessi di questa estensione per Prolog che, nello sviluppo di un sistema per gestire le consegne di fiori mediante fattorini nella città di New York, mapparono la città per quartieri (Brooklyn, Queens, etc…), e il cui esempio ne è stato di fonte ispirazione.

Qui di seguito andrò a descrivere in maniera formale le varie regioni del campo di calcio utilizzate per il riconoscimento dell'evento Cross, nonché per stabilire l'area di interesse coinvolta per un altro evento complesso, il tiro-salvato (SavedShot) per il quale era

necessario specificare dove avveniva il tiro. Le aree di interesse sono quattro: goal-area, penalty-area, side, inner-pitch. Qui nel seguito uno schema rappresentativo.

Postprocessamento

Così come è necessario per Etalis avere una fase di preprocessamento che garantisce la trasformazione dei dati in ingresso nel suo formato proprietario, è necessaria una fase di postprocessamento per emettere la lista di eventi complessi in un formato dati condiviso e consolidato. Questo compito è necessario perché i risultati del sistema di riconoscimento vanno confrontati con il ground truth, ossia con la lista di eventi complessi effettivamente avvenuti. La fase di confronto dei risultati e la valutazione di essi sarà descritta più avanti ma sarà fondamentalmente il motivo per cui questo compito di trasformazione del formato dati è necessario.

Per eseguire questo processamento è stato creato un modulo software che è del tutto speculare rispetto a quanto fatto per la fase di preprocessamento. Ossia è stato scritto con il linguaggio di programmazione Python, alla quale è stata abbinata una interfaccia applicativa chiamata *ElementTree API*.

5.3 Interfaccia di esportazione eventi dal simulatore di gioco

Questo paragrafo descrive in che modo è stata costruita l'interfaccia che viene usata dal simulatore di gioco per generare la lista di eventi che si sono verificati. Per questo processo si è usata la libreria Boost C++ per fare serializzazione, e a valle è stato effettuato un passo di normalizzazione.

Serializzazione

Si tratta di andare ad utilizzare il modulo di serializzazione della libreria Boost C++, libreria già presente ed integrata nel simulatore di gioco, e che non avrebbe richiesto nessuna modifica alla configurazione del gioco stesso. Fattore molto importante in quanto critico per sforzo e compatibilità.

La serializzazione è il processo di trasformare un oggetto in memoria in un flusso di byte in modo da poter fare operazioni come per esempio salvarlo su disco o inviarlo via rete. La deserializzazione è il processo inverso: trasformare un flusso di byte in un oggetto in memoria.

La serializzazione mediante libreria C++ Boost consente di vuotare (eseguire un *dump*), in maniera semplice ed automatica, una struttura dati contenuta in memoria direttamente su un supporto locale chiamato *archivio*. Quest'ultimo potrà poi essere caricato in memoria in un secondo momento (eseguire un *restore*), con l'obiettivo per esempio di confrontare dei valori, alcuni di questi memorizzati in passato su un archivio con degli altri che sono stati generati da un nuovo flusso di esecuzione. Le operazioni base sono pertanto esprimibili mediante due operazioni base: salvataggio e caricamento.

Esempio di operazioni base

save_program.cpp	load_program.cpp

#include <boost/archive/text_oarchive.hpp>
#include <iostream>
#include <fstream>
void save()
{
std::ofstream file("archive.txt");
boost::archive::text_oarchive oa(file);
std::string s = "Hello World!\n";
oa << s;
}
int main()
{
save();
	#include <boost/archive/text_iarchive.hpp>
	#include <iostream>
	#include <fstream>
	void load()
	{
	std::ifstream file("archive.txt");
	boost::archive::text_iarchive ia(file);
	std::string s;
	ia >> s;
	std::cout << s << std::endl;
	}
	int main()
	{

}	load(); }

La serializzazione può avvenire per strutture dati base o astratte, ossia per classi o puntatori a classi, vettori, ma anche per collezioni di oggetti della libreria standard (STL). E' importante specificare che la serializzazione per classi create in maniera proprietaria, e quindi manuale, avviene grazie ad un *template method* che va implementato nella classe e specifica quali membri della classe vanno serializzati e in che modo.

Esempio di serializzazione per tipi o classi proprietarie

Type definition	Template method

typedef struct date {
unsigned int m_day;
unsigned int m_month;
unsigned int m_year;
} date;
	template<class Archive>
	void serialize(Archive& archive, const unsigned int version)
	{
	archive & BOOST_SERIALIZATION_NVP(m_day);
	archive & BOOST_SERIALIZATION_NVP(m_month);
	archive & BOOST_SERIALIZATION_NVP(m_year);
	}

Quello che si è andato a fare è stato quello di definire numerose classi proprietarie, in maniera manuale ma in modo da garantire piena flessibilità e personalizzazione, con l'obiettivo di ricreare mediante serializzazione lo stesso schema XSD corrispondente al file esportato in CVAT. Il file esportato da CVAT ha chiaramente una gerarchia di classi e, il modulo di serializzazione della libreria Boost ne ha il pieno supporto, tant'è che il processo di ricostruzione della gerarchia delle classi non è stato complicato ma soltanto laborioso. Infatti, nel momento in cui il programma richiede la serializzazione di una classe (richiamando il *template method serialize* già discusso sopra) e all'interno non vi è soltanto la serializzazione dei tipi base ma anche di oggetti, creati come istanza di altri classi proprietarie, allora in maniera ricorsiva verranno richiamati i *template method* di questi oggetti.

Normalizzazione

Sebbene il modulo di serializzazione della libreria Boost C++ è in grado di gestire propriamente i vettori, ossia una sequenza di dati organizzata in maniera vettoriale, non è capace di esportarli nell'esatto formato che è richiesto da CVAT. Inoltre, la serializzazione degli oggetti mediante Boost presenta, nel file esportato, alcuni

attributi aggiuntivi e non voluti, ma che sono generati al momento della instanziazione delle classi e quindi insiti negli oggetti. Esempi di attributi di questo tipo sono lo object_id, che vanno a "sporcare" il file esportato e che impediscono allo strumento di annotazione di poter importare il file e caricare le informazioni. Pertanto per eliminare questi difetti, è stato introdotto un altro passo di processamento che, dopo la fase di serializzazione, va a correggere e normalizzare il formato a quello richiesto da CVAT. Il nome di questo passaggio è stato pertanto fissato a *normalizzazione*. Il passo di normalizzazione è stato eseguito con una libreria, pugixml, di tipo multipiattaforma, e rilasciata con licenza open-source e per la quale è disponibile una buonissima documentazione. In generale, quindi, questo passaggio va a ultimare e perfezionare l'esportazione dei dati per ottenere il file con formato voluto. Infatti, mentre il processo di serializzazione agisce in maniera più corposa all'esportazione dei dati, la normalizzazione ne è uno passaggio finale. La libreria agisce come XML parser, ed è pertanto in grado di analizzare un documento in formato dati XML, modificarlo e infine esportarlo.

Caratteristiche della libreria

Il documento XML è rappresentato secondo una struttura dati ad albero dove, partendo dal nodo radice che è il documento stesso, ci possono essere nodi figlio, ognuno dei quali può avere una collezione di nodi figlio oppure una collezione di attributi.

Esempio
```
<node attr="value"><child/></node>
```

Per quei nodi che presentano un contenuto testuale di tipo *plain-text*, è stato definito un tipo particolare di nodo chiamato *plain-character data node* o abbreviato PCDATA node. Per questo tipo di nodi esistono dei metodi specifici per la manipolazione (lettura e scrittura) del contenuto testuale, che si differenziano dai primi, ossia i nodi base.

Esempio
```
<node> text1 <child/> text2 </node>
```

Per accedere ad un nodo, a partire dal nodo radice, è possibile procedere con meccanismi diversi:

- Operazioni base di attraversamento
- Eseguire una ricerca
- Fare un ciclo iterativo di attraversamento
- Eseguire una discesa ricorsiva di attraversamento

Alcune delle operazioni base di attraversamento sono:

```
xml_node xml_node::parent() const;
xml_node xml_node::first_child() const;
xml_node xml_node::last_child() const;
xml_node xml_node::next_sibling() const;
```

Oppure e' possibile specificare il nome di un nodo per eseguirne una ricerca, dove in questo caso ne verrà restituita la prima occorrenza:

```
xml_node xml_node::child(const char_t* name) const;
```

E' possibile eseguire un ciclo iterativo di attraversamento sui nodi o sugli attributi di una lista facendo uso degli iteratori.

```
class xml_node_iterator;
typedef xml_node_iterator xml_node::iterator;
iterator xml_node::begin() const;
iterator xml_node::end() const;
```

Oppure è anche possibile eseguire una discesa ricorsiva di attraversamento.

```
class xml_tree_walker
{
public:
   virtual bool begin(xml_node& node);
   virtual bool for_each(xml_node& node) = 0;
   virtual bool end(xml_node& node);

   int depth() const;
};

bool xml_node::traverse(xml_tree_walker& walker);
```

Una volta fatto accesso al nodo desiderato oppure all'attributo desiderato, arriva il momento della lettura oppure della modifica con l'operazione di scrittura. In tal senso abbiamo a disposizione:

- I metodi base per assegnazione sui nodi
- I metodi base per assegnazione su attributi
- La possibilità di appendere nuovi nodi a quelli esistenti
- La possibilità di rimuovere dei nodi
- La possibilità di copiare dei nodi
- La possibilità di spostare dei nodi

Infine l'operazione di salvataggio ed esportazione del documento in formato dati XML avviene con dei metodi di servizio.

Capitolo 6

Testing e validazione

6.1 Strumenti di testing

Passo anch'esso di notevole importanza, nel ciclo di sviluppo del software, è quello del testing. Questa fase si è caratterizzata di un controllo sulla congruenza delle regole implementate, sulla corrispondenza semantica degli eventi tra il ground truth e quelli riconosciuti, ed infine un perfezionamento dei parametri del sistema.
Tuttavia per raggiungere questo scopo, non è bastato utilizzare il sistema di riconoscimento stesso ma è stato necessario sviluppare un sistema complementare, di visualizzazione, per discriminare visivamente l'occorrere degli eventi e le loro dinamiche. Sebbene sia possibile tenere in considerazione lo stream di output del sistema ETALIS, che raccoglie sia gli eventi atomici che gli eventi complessi e permette di osservare sulla base di quali eventi atomici sono stati emessi gli eventi complessi, esso non basta perché è uno strumento fine, di dettaglio, ma poco pratico per il fine del testing. Quando si osserva lo stream di ETALIS si sta osservando la lista in un formalismo poco leggibile e in un file di testo, poco maneggiabile e poco significativo se è necessario una visione di alto livello. Esso non permette, infatti, di sapere quale è stata la dinamica dell'evento che si sta osservando. Vediamo un esempio in Fig.31.

```
*Event: ballPossession(75,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,8),datime(2019,9,18,18,29,32,8)]
*Event: ballPossession(85,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,9),datime(2019,9,18,18,29,32,9)]
*Event: ballPossession(95,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,10),datime(2019,9,18,18,29,32,10)]
*Event: ballPossession(105,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,11),datime(2019,9,18,18,29,32,11)]
*Event: ballPossession(115,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,12),datime(2019,9,18,18,29,32,12)]
*Event: ballPossession(125,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,13),datime(2019,9,18,18,29,32,13)]
*Event: ballPossession(135,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,14),datime(2019,9,18,18,29,32,14)]
*Event: ballPossession(145,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,15),datime(2019,9,18,18,29,32,15)]
*Event: ballPossession(155,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,16),datime(2019,9,18,18,29,32,16)]
*Event: ballPossession(165,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,17),datime(2019,9,18,18,29,32,17)]
*Event: ballPossession(175,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,18),datime(2019,9,18,18,29,32,18)]
*Event: ballPossession(185,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,19),datime(2019,9,18,18,29,32,19)]
*Event: ballPossession(195,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,20),datime(2019,9,18,18,29,32,20)]
*Event: ballPossession(205,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,21),datime(2019,9,18,18,29,32,21)]
*Event: ballPossession(215,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,22),datime(2019,9,18,18,29,32,22)]
*Event: ballPossession(225,5,0,55.0,36.3,20,1,82.5,32.2) @ [datime(2019,9,18,18,29,32,23),datime(2019,9,18,18,29,32,23)]
*Event: kickingTheBall(241,5,0,55.0,36.3) @ [datime(2019,9,18,18,29,32,24),datime(2019,9,18,18,29,32,24)]
*Event: ballPossession(263,8,0,54.9,33.2,21,1,82.3,39.7) @ [datime(2019,9,18,18,29,32,25),datime(2019,9,18,18,29,32,25)]
*Event: pass(241,263,5,0,8) @ [datime(2019,9,18,18,29,32,24),datime(2019,9,18,18,29,32,25)]
*Event: ballPossession(273,8,0,54.9,33.6,21,1,82.1,39.7) @ [datime(2019,9,18,18,29,32,26),datime(2019,9,18,18,29,32,26)]
*Event: ballPossession(283,8,0,54.9,34.0,21,1,81.8,39.8) @ [datime(2019,9,18,18,29,32,27),datime(2019,9,18,18,29,32,27)]
*Event: ballPossession(293,8,0,55.0,34.4,21,1,81.4,39.9) @ [datime(2019,9,18,18,29,32,28),datime(2019,9,18,18,29,32,28)]
*Event: ballPossession(303,8,0,55.1,34.9,21,1,80.9,39.9) @ [datime(2019,9,18,18,29,32,29),datime(2019,9,18,18,29,32,29)]
*Event: ballPossession(313,8,0,55.2,35.4,21,1,80.4,40.0) @ [datime(2019,9,18,18,29,32,30),datime(2019,9,18,18,29,32,30)]
*Event: ballPossession(323,8,0,55.3,35.8,21,1,79.9,40.1) @ [datime(2019,9,18,18,29,32,31),datime(2019,9,18,18,29,32,31)]
*Event: ballPossession(333,8,0,55.5,36.1,21,1,79.3,40.2) @ [datime(2019,9,18,18,29,32,32),datime(2019,9,18,18,29,32,32)]
*Event: ballPossession(343,8,0,55.8,36.4,21,1,78.8,40.3) @ [datime(2019,9,18,18,29,32,33),datime(2019,9,18,18,29,32,33)]
*Event: ballPossession(353,8,0,56.2,36.7,21,1,78.2,40.4) @ [datime(2019,9,18,18,29,32,34),datime(2019,9,18,18,29,32,34)]
*Event: ballPossession(363,8,0,56.6,37.0,21,1,77.5,40.5) @ [datime(2019,9,18,18,29,32,35),datime(2019,9,18,18,29,32,35)]
```

Figura 19 - Esempio di stream di eventi emessi da Etalis.

Come si vede dall'immagine è possibile sapere che l'evento Pass è stato emesso dopo l'occorrere di un calcio palla e un controllo palla, ma non è possibile vedere come è effettivamente avvenuto questo passaggio. Pertanto è stato necessario pensare, progettare e sviluppare un sistema di visualizzazione.

Sistema di visualizzazione

Il sistema di visualizzazione è uno strumento di debug e testing, progettato con l'idea di analizzare in maniera visuale gli insiemi di dati generati durante il lavoro di tesi. L'analisi è un passo necessario per garantire la coerenza e verificare la correttezza degli eventi presenti.

L'interfaccia utente si compone di una finestra che la libreria OpenCV mette a disposizione per la visualizzazione a schermo di un immagine, ossia nel nostro caso un frame, con in più la trackbar e altre funzionalità per l'interazione descritte in seguito. Nel frame selezionato vengono caricati i BBox dei giocatori e della palla e questi sono rappresentati a schermo.

Figura 20 - Esempio di UI del sistema di visualizzazione.

Lista eventi mostrati

L'interfaccia mostra le liste degli eventi (ground truth e detected, atomici e complessi). Essi sono mostrati in basso, (come mostrato in Fig. 21) mediante delle delle righe di testo, e mostrano in maniera precisa e puntuale il nome dell'evento e i vari attributi associati, frame per frame.

Gli eventi atomici mostrati sono:

- BallPossession
- KickingTheBall
- Tackle
- BallDeflection
- BallOut
- Goal
- Foul
- Penalty

Gli eventi complessi mostrati sono:

- Pass
- Tackle
- PassThenGoal
- FilteringPass
- FilteringPassThenGoal
- Cross
- CrossThenGoal
- Shot
- ShotThenGoal
- SavedShot

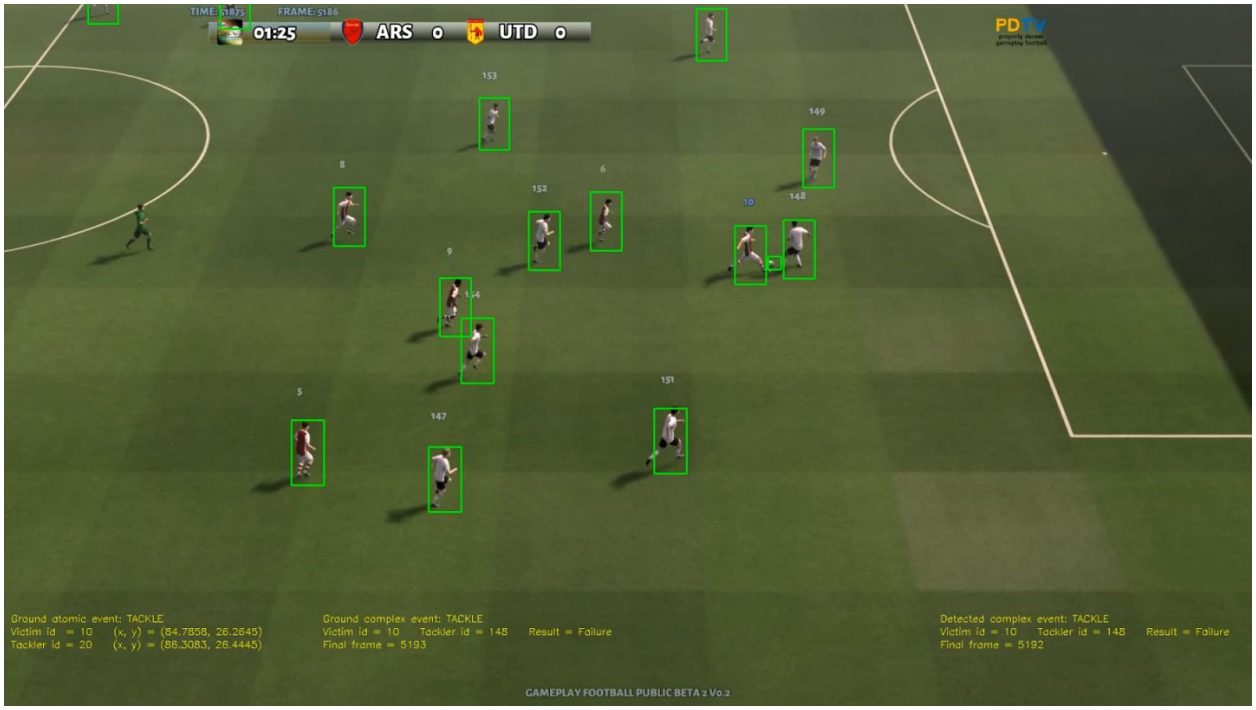


Figura 21 - Esempio di frame con annotazioni in sovraimpressione.

Come specificato nel capitolo “Dati utilizzati” la video registrazione ha un frame rate di 25 fps. Tuttavia, poiché il motore di gioco ha un frame-rate di 100 fps, per ogni frame della registrazione ci siano 4 eventi. Allora è necessario mostrarli e indicare in maniera puntuale a quale frame del gioco essi fanno riferimento.

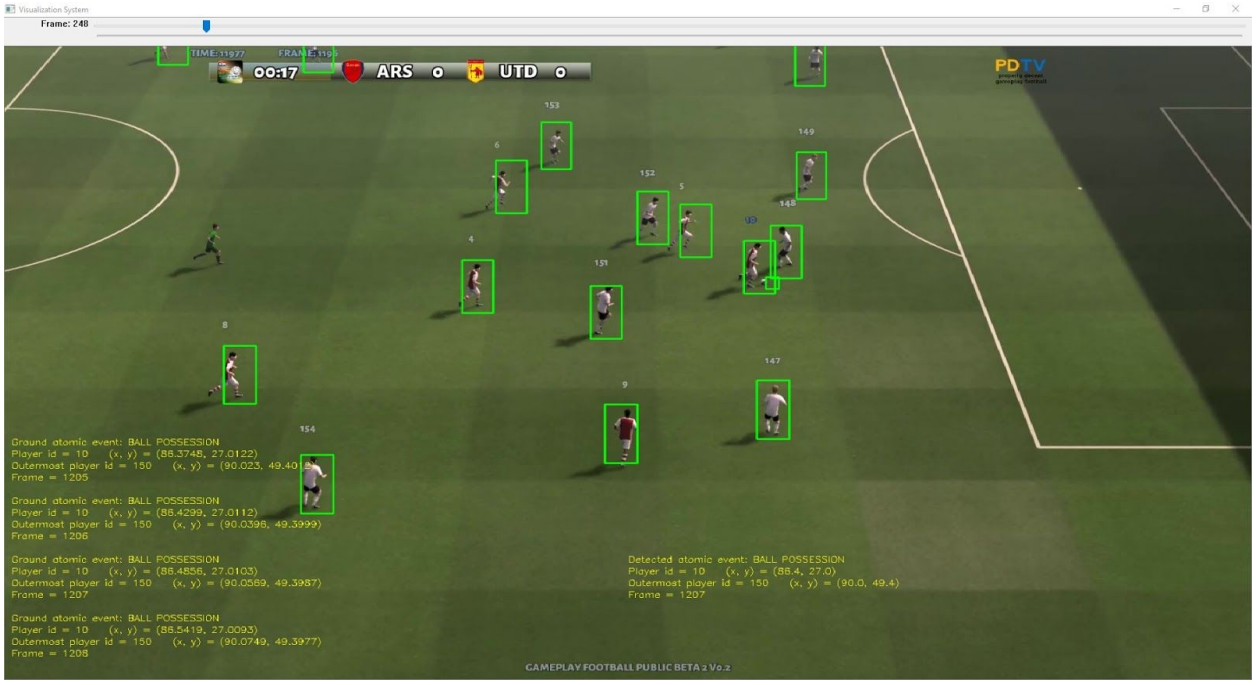


Figura 22 - Esempio di annotazioni per i quattro frame di gioco per singola immagine in sovraimpressione.

Interfaccia di interazione

Il sistema di visualizzazione offre due meccanismi di interazione che garantiscono la navigabilità del video e il passaggio tra i vari frame: un trackbar e due tasti funzione.

La *trackbar* è stata resa disponibile in alto, nell'interfaccia utente, e permette di selezionare il frame di interesse. La trackbar offre la possibilità di muoversi in maniera rapida e accessibile tra due punti molto distanti della video registrazione.

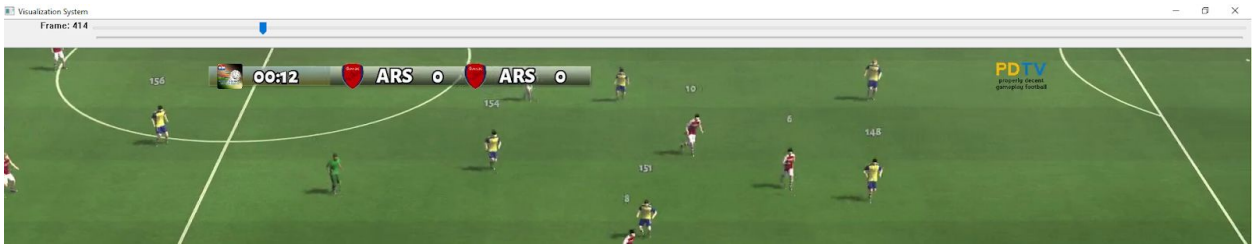


Figura 23 - Trackbar di selezione per i frame di interesse.

Sono disponibili *i tasti direzionali avanti e indietro* della tastiera che consentono di navigare il video in maniera più fine una volta selezionata una zona di interesse con la trackbar.

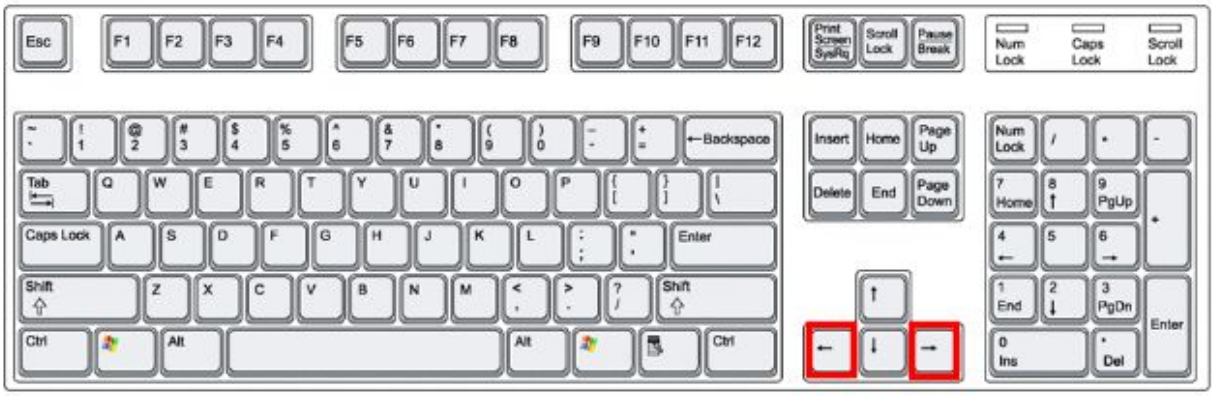


Figura 24 - Frecce direzionali per una navigazione fine tra i frame del video.

Il tasto funzione *barra spaziatrice* consente in maniera alternativa di mandare in play o in pausa la registrazione, e di visualizzare la dinamica del match sportivo un frame per secondo. Questa funzione garantisce quindi la fruizione continua dei frame del flusso video, in quanto in alcune situazioni non è richiesta un'analisi attenta di quei frame.

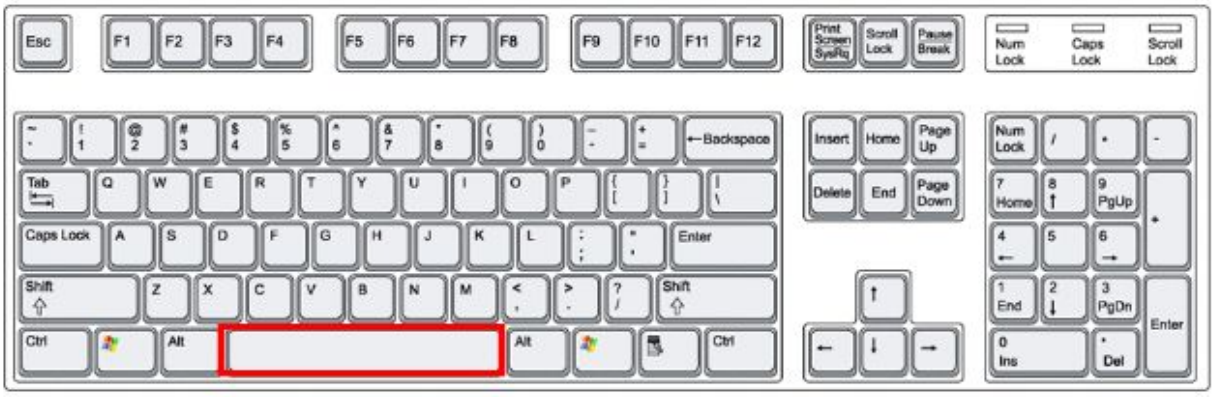


Figura 25 - Barra spaziatrice per mandare in riproduzione il video, e consentire uno scorrimento continuo della porzione di video di interesse.

Per terminare la fruizione del video e chiudere il sistema di visualizzazione è possibile premere il tasto *Esc*, che è azionabile in qualsiasi momento.

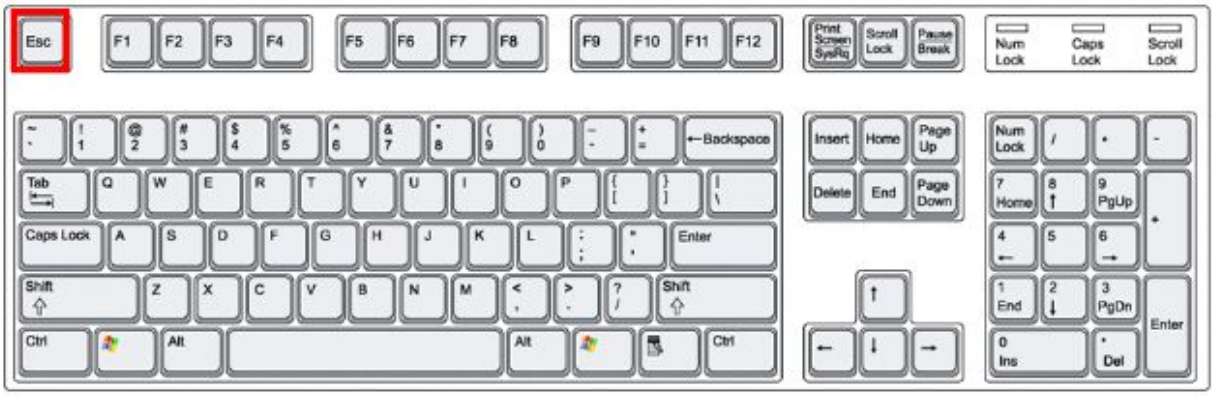


Figura 26 - Tasto Esc per terminare il sistema di visualizzazione e la fruizione del video.

Esportazione come video

Il sistema permette di visualizzare i BBox dei giocatori e della palla per rappresentare le loro posizioni, di aggiungere le annotazioni degli eventi su ciascun frame, e di visualizzarle a video per una corretta analisi di quali eventi sono avvenuti e quali sono stati rilevati. Inoltre, per garantire la massima compatibilità con i vari sistemi (PC, Tablet, Smartphone) e i vari sistemi operativi (Linux, Windows, MacOs) e, dato che si è fatto il lavoro in un ambiente composto da un team, per il fine di fornire la massima facilità di fruizione, si è pensato di creare un tool integrato al sistema di visualizzazione in grado di esportare un video finale. Il video in questione è l'unione delle immagini video-registrate dal simulatore di gioco e, in più, le annotazioni del sistema di riconoscimento, le quali sono rappresentate come già descritto in

sovraimpressione. L'idea è di esportare un video che contenga tutta la dinamica del gioco, con le posizioni dei giocatori, i Bounding Box, le indicazioni sul passare del tempo, ossia i frame del gioco e il frame relativo all'inizio della partita, e le annotazioni degli eventi distinte per ground truth e detected, e atomici e complessi. Il filmato prodotto un un video in formato mp4, con codifica MPEG-4, esportato a 25 fps. Questi parametri sono stati scelti per garantire coerenza con il file dato in ingresso al sistema.

Modalità di esportazione

Per esportare il video e ottenere il file in output basta lanciare il sistema di visualizzazione, e premere il tasto funzione W: inizializza il processo e genera così il video di output.

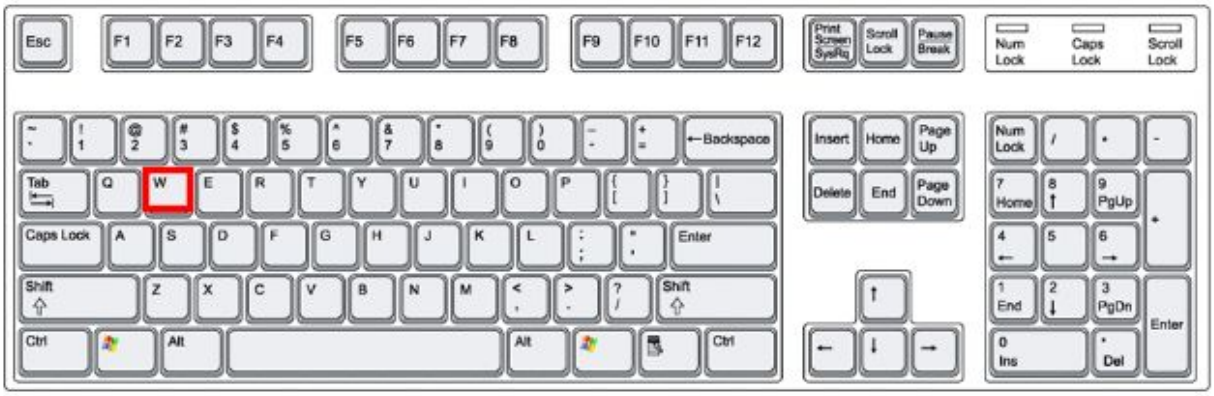


Figura 27 - Tasto per inizializzare l'esportazione video.

Sistema di visualizzazione

Il toolkit necessario allo sviluppo del sistema di visualizzazione comprende: il videogioco *GameplayFootball* per la generazione dei dati di natura sintetica, la libreria software open-source *OpenCV* per la grafica dei computer che consente di manipolare i contenuti multimediali in formato video e immagini, e lo strumento di registrazione dello schermo *OBS*.

Dati in ingresso

I dati che sono stati utilizzati dal sistema sono: una videoregistrazione, che contiene il filmato video da mostrare; i dati posizionali, da sfruttare per mostrare i BBox dei giocatori; il ground truth degli eventi atomici e complessi e gli eventi riconosciuti atomici e complessi, che servono per mostrare gli eventi in sovraimpressione.

Dati posizionali

Si tratta dei dati posizionali dei giocatori e della palla esportati insieme al tempo di gioco e al timestamp in millisecondi, relativo rispetto all'inizio della partita. Ogni riga del file contiene rispettivamente le seguenti informazioni:

1) Millisecondi relativi al calcio di inizio;
2) Frame del gioco;
3) Id giocatore, o 128 per la palla;
4) Coordinata x sullo schermo;
5) Coordinata y sullo schermo;

Secondo il formato:
<timestamp_relativo> <tempo_di_gioco> <ID> <pos_x> <pos_y>

Esempio
30199 3018 128 883 441 30199 3018 0 0 362 30199 3018 1 490 96 […] 30199 3018 155 173 383 30199 3018 156 613 36 30204 3019 128 881 442

30204 3019 0 0 362
30204 3019 1 488 96

Ground truth atomici e complessi

I dati degli eventi atomici e complessi così come vengono esportati dal simulatore di gioco, o meglio, dall'interfaccia che è stata sviluppata dal gruppo di ricerca per la loro esportazione.

```
<track id="750" label="BallPossession">
    <box frame="1135" keyframe="1" occluded="0" outside="0" xbr="1.24157999999999
        <attribute name="playerId">10</attribute>
        <attribute name="teamId">0</attribute>
        <attribute name="outermostOtherTeamDefensivePlayerId">22</attribute>
        <attribute name="outermostOtherTeamDefensivePlayerTeamId">1</attribute>
    </box>
</track>
<track id="751" label="BallPossession">
    <box frame="1136" keyframe="1" occluded="0" outside="0" xbr="1.24157999999999
        <attribute name="playerId">10</attribute>
        <attribute name="teamId">0</attribute>
        <attribute name="outermostOtherTeamDefensivePlayerId">22</attribute>
        <attribute name="outermostOtherTeamDefensivePlayerTeamId">1</attribute>
    </box>
</track>
<track id="752" label="BallPossession">
    <box frame="1137" keyframe="1" occluded="0" outside="0" xbr="1.24157999999999
        <attribute name="playerId">10</attribute>
        <attribute name="teamId">0</attribute>
        <attribute name="outermostOtherTeamDefensivePlayerId">22</attribute>
        <attribute name="outermostOtherTeamDefensivePlayerTeamId">1</attribute>
    </box>
</track>
```

Figura 28 - Esempio di eventi BallPossession con annessi attributi.

```
<track id="3" label="Pass">
    <box frame="894" keyframe="1" occluded="0" outsi
        <attribute name="sender">4</attribute>
        <attribute name="teamId">0</attribute>
        <attribute name="receiver">10</attribute>
    </box>
    <box frame="895" keyframe="0" occluded="0" outsi
        <attribute name="sender">4</attribute>
        <attribute name="teamId">0</attribute>
        <attribute name="receiver">10</attribute>
    </box>
    <box frame="896" keyframe="0" occluded="0" outsi
        <attribute name="sender">4</attribute>
        <attribute name="teamId">0</attribute>
        <attribute name="receiver">10</attribute>
    </box>
```

Figura 29 - Esempio di evento Pass con annessi attributi

Eventi riconosciuti atomici e complessi

Gli eventi riconosciuti atomici e complessi hanno lo stesso formato riportato per ground truth atomici e complessi.

Dati in uscita

Il sistema è in grado di unire i dati in ingresso, ed elaborarli frame dopo frame fintantoché non si ha in uscita un file video. La video-registrazione ottenuta con lo strumento sopramenzionato OBS è un filmato video multimediale in risoluzione Full-HD, in formato file MKV e codifica H264, a 25 fps.

Workflow operativo

Qui andrò a spiegare i passi che vengono compiuti dal software per garantire la visualizzazione dei dati e la loro interazione.

1. Dal video ai frame

Il primo passo consiste nel processare la videoregistrazione ed esportare localmente, su disco locale, frame per frame. Questo garantisce l'accesso diretto al frame di interesse qualora l'utente finale necessiti la visualizzazione, senza salvare in memoria primaria l'intero filmato che risulterebbe in una degenerazione dello spazio in memoria.

2. Caricamento delle posizioni

Vengono caricate in memoria le posizioni dei giocatori e della palla, così che, quando viene richiesto l'interesse per uno specifico frame, vengono visualizzate a schermo queste posizioni sotto forma di Bounding Box.

3. Caricamento degli eventi

Vengono caricati in memoria gli eventi di interesse coinvolti in quel frame selezionato, distinguendoli per atomici e complessi, e ground truth o detected.

4. Utilizzo da parte dell'utente

A questo punto l'utente può usare o la trackbar presente in alto, oppure le frecce direzionali per spostarsi nel video, frame per frame. Quando l'utente si sposta di posizione, il software, che precarica una finestra di file dal disco in memoria principale, sposta la finestra e precarica altri frame. In questo modo il caricamento è molto rapido e non richiede attese.

Capitolo 7

Risultati

I risultati ottenuti dal sistema di riconoscimento, ossia la lista di eventi complessi, sono stati poi oggetto di valutazione perché il fine ultimo è costruirne una misura di precision e recall ed f1-score, le tre *metriche di valutazione* tenute in considerazione. Gli eventi, infatti, sono stati processati da un algoritmo di valutazione che, in essenza, confronta gli eventi riconosciuti dal sistema con quelli generati dal simulatore di gioco (ground truth), assicurandosi che quelli riconosciuti siano presenti in quelli originali e viceversa. I risultati ottenuti dall'algoritmo di confronto sono poi stati rappresentati, mediante tecniche di visualizzazione dei dati, con grafici a barre per incrementarne la comprensione e la percezione.

Dataset utilizzato

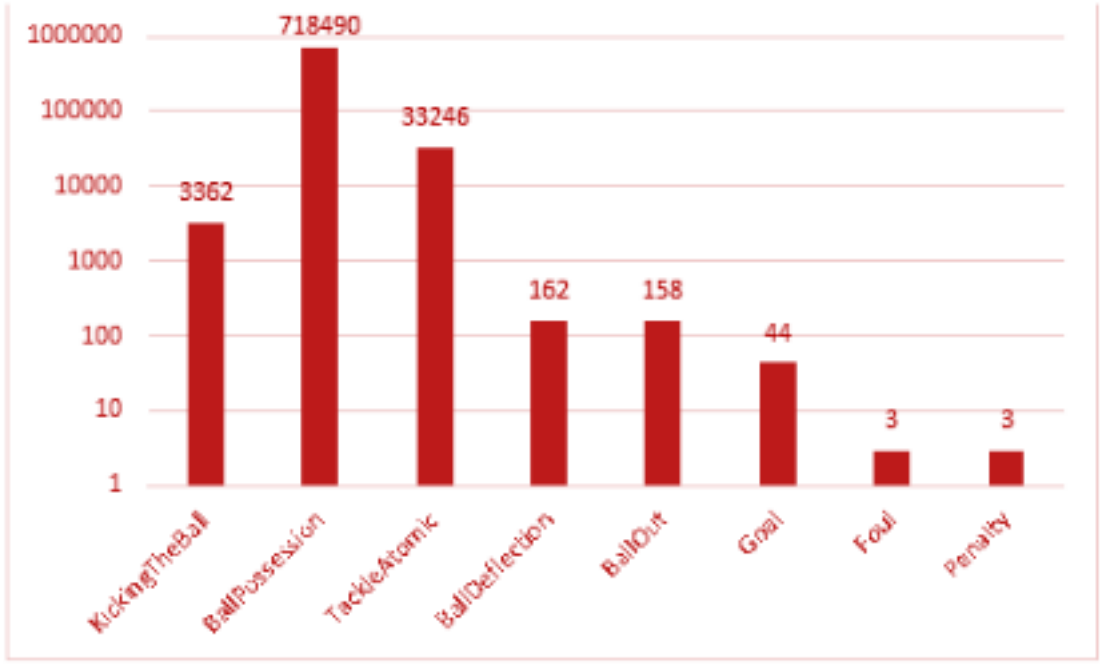
I risultati qui riportati sono stati generati a partire da un dataset di 500 minuti, generato con il simulatore di gioco e diviso successivamente in validation set e testing set. Il numero totale di eventi è 1591199, mentre il numero totale di eventi complessi è 8633. Qui di seguito riporto lo split del dataset in validation e testing set.

Tipo	**Tempo di gioco**	**Num. eventi atomici**	**Num. eventi complessi**
Validation	250 minuti	755468	4430
Testing	250 minuti	827098	4203

Il validation set è stato utilizzato per verificare che il sistema di inferenza costruito sulla base del sistema di regole riuscisse ad emettere gli eventi desiderati e nelle modalità richieste, ma non solo, anche per verificare le soglie e i parametri del sistema. Mentre il testing set è stato utilizzato per la generazione dei risultati.

Training set

La numerosità degli eventi nel validation set è rappresentata in figura 30.



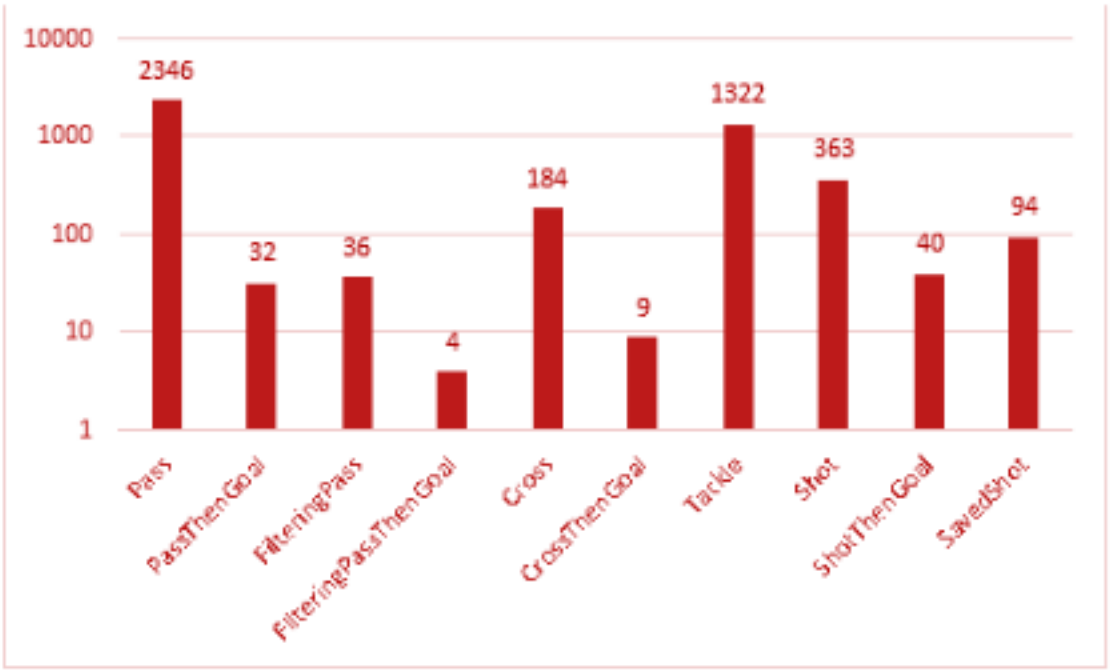


Figura 30 - Eventi presenti nel validation set.

Testing set

La numerosità degli eventi nel testing set è rappresentata in figura 31.

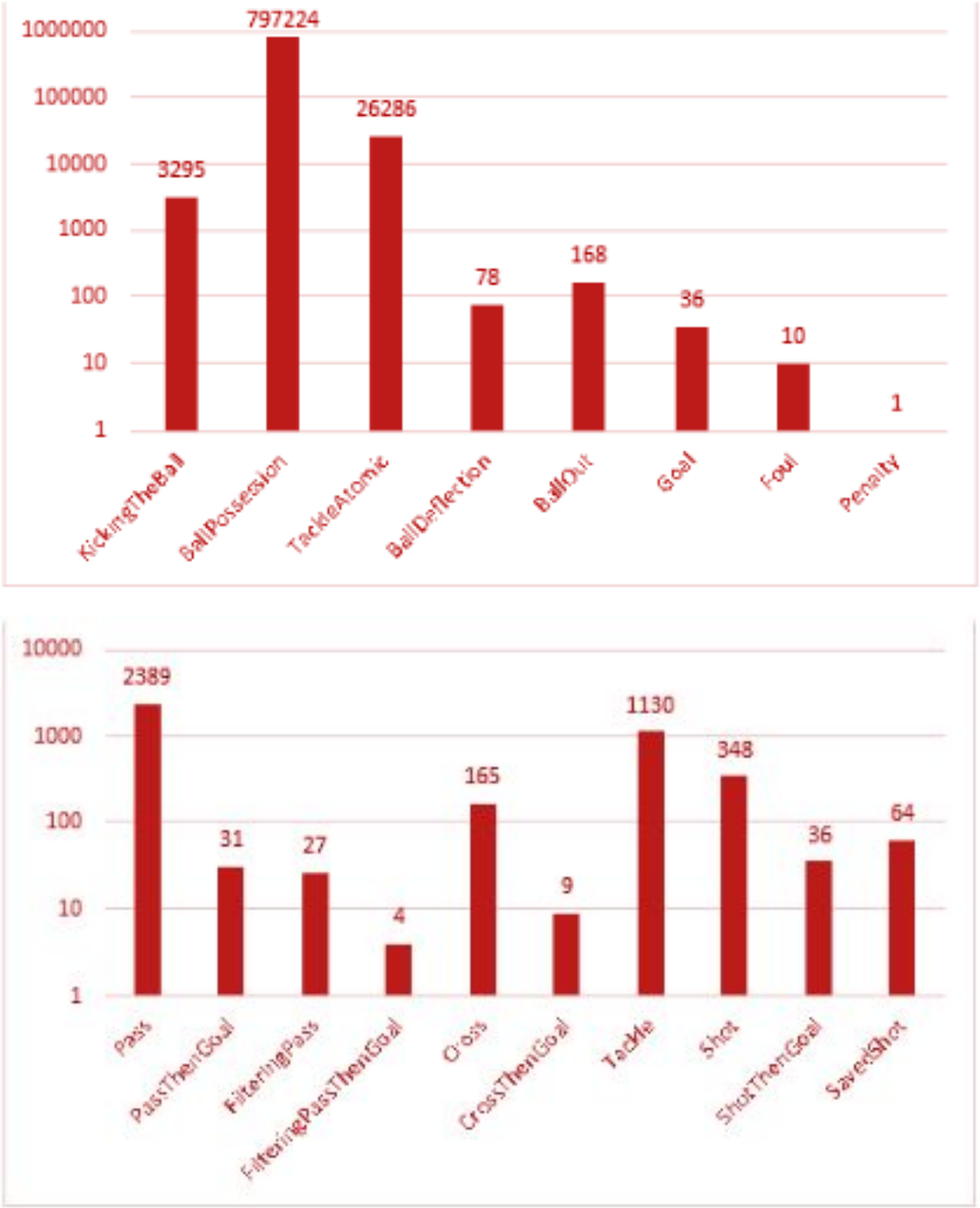


Figura 31 - Eventi presenti nel testing set.

Metriche di valutazione

Come metriche sono state utilizzate precision, recall, e f-score relative a ciascuno degli eventi. L'interpretazione per il valore di precisione e recall è la seguente:

- **Precision**: un valore basso dice che il sistema di riconoscimento non emette eventi effettivamente occorsi ma che sono presenti nel ground truth.

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

- **Recall**: un valore basso indica che il sistema di riconoscimento trova molti eventi che non sono presenti nel ground truth.

$$recall = \frac{true\ positives}{true\ positives\ + false\ negatives}$$

- **F-score**: metrica con valori aggregati sui due precedenti.

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

Algoritmo di valutazione

L'algoritmo sviluppato dal collega di tesi, ispirato a quanto descritto in [18] scorre gli eventi del ground truth e controlla che nel suo intervallo temporale ci sia l'inizio di uno degli eventi rilevati. Se lo trova, ne viene calcolata l'unione e l'intersezione dei due. Due eventi si dicono corrispondenti se il rapporto tra intersezione e unione è superiore al 20%.

Esempio

Il procedimento confronta gli eventi del ground truth e gli eventi rilevati. Essi vengono messi a confronto calcolandone unione ed intersezione. Per gli eventi 1, 4 e 5 si sono calcolati i valori di union-intersection, ma l'evento 5 viene classificato come non riconosciuto per via del valore di ratio union-intersection inferiore al 20%. Per gli eventi 2 e 3 invece non vengono calcolati i valori union-intersection perché gli eventi 2 sono di due classi differenti (eventi diversi) mentre gli eventi 3 non hanno istanti di tempo in comune.

Figura 32 - Esempio di procedimento per la valutazione dei risultati.

Risultati finali

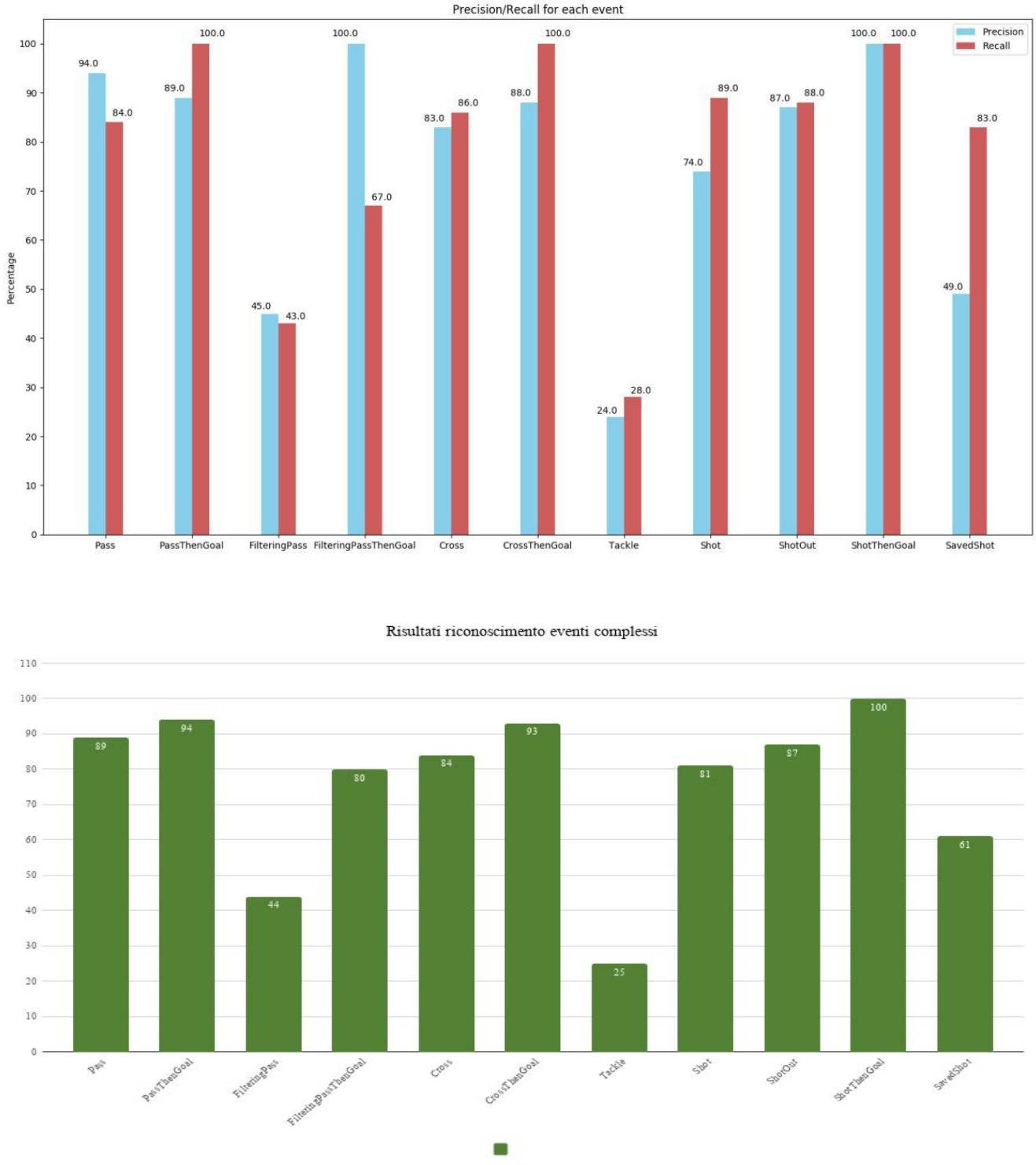


Figura 33 - Risultati del sistema di riconoscimento per eventi complessi, con il testing set.

Dai risultati che emergono dal sistema di riconoscimento per eventi complessi si evince che, su 11 classi di eventi complessi, per 8 di essi si hanno valori di f1-score superiori all'80%, mentre per tre di essi si hanno valori più scarsi, intorno al 50% di f1-score. In particolare gli eventi Tackle e SavedShot riflettono alcuni problemi già emersi nel lavoro del collega che si è occupato della parte del sistema di riconoscimento degli eventi atomici, il quale evidenziava un limite nella capacità di riconoscimento di quegli eventi a partire dai soli dati posizionali. Gli eventi Tackle e SavedShot dipendono da due eventi atomici Tackle e BallDeflection, e ne sono una loro combinazione. Se vi sono difficoltà nel riconoscimento di questi eventi a livello atomico, non si riesce a inferire nulla al livello dei complessi.
Tuttavia il risultato è positivo perché se confrontato con la letteratura si vede che il numero di eventi che vengono riconosciuti in questo lavoro di tesi è maggiore e con performance superiori. In particolare, se si mettono a confronto le altre soluzioni basate su dati posizionali il solo evento che si può confrontare è il passaggio, rappresentato in tabella 8, per il quale si è raggiunto un valore di f-score pari a 89% contro il 50% della letteratura. Se si considerano altre soluzioni non basate su dati posizionali ma sempre su sistemi di regole, il passaggio raggiunge valori di f-score pari a 60%, sempre inferiore a 89%. Un altro evento confrontabile è il tiro che assume valori di 81% di f-score contro il 50% della letteratura.

Evento	**TP**	**FP**	**TN**
Pass	2046	128	380
PassThenGoal	31	4	0
FIlteringPass	10	12	13
FilteringPassThenGoal	2	0	1
Cross	152	32	24
CrossThenGoal	7	1	0
Tackle	326	1038	829
Shot	204	70	26
ShotOut	114	17	16
ShotThenGoal	36	0	0
SavedShot	53	56	11

Tabella 7 - Dati associati al grafico di figura 33.

Risultato	Precision	Recall	F-score
Risultati della soluzione proposta	94%	84%	89%
“Recognizing Compound Events in Spatio-Temporal Football Data” [16]	43%	64%	51%
“Soccer event recognition technique based on pattern matching” [17]		60%	

Tabella 8 - Confronto della soluzione proposta con le soluzioni in letteratura rispetto all’evento passaggio semplice.

Capitolo 8

Conclusioni

Il lavoro di tesi si è incentrato sul riconoscimento degli eventi sportivi e ha visto sin dalla loro formalizzazione, fino al loro riconoscimento, lo sviluppo di una moltitudine di strumenti: sistema di riconoscimento, strumenti di valutazione, algoritmi di ottimizzazione, sistemi di visualizzazione.

La tassonomia iniziale prefissata era molto ampia e su un totale di 21 eventi di interesse 17 di essi vengono classificati correttamente con un f-score superiore al 80%, e tre superiori al 90%. Per gli altri 4 eventi non si è reso possibile raggiungere valori alti per limitatezza semantica legata ai dati posizionali.

Una prima forma di miglioramento potrebbe essere quella di utilizzare un dataset con dati posizionali non ottenuti mediante generazione sintetica, ma con un qualche altro meccanismo che possa per esempio essere setup di telecamere. La generazione sintetica comporta inevitabilmente un limite di realismo e possibili imprecisioni.

Un altro miglioramento possibile o altra strada percorribile è sicuramente quella di utilizzare dataset di natura diversa, basati su immagini, e intraprendere con un approccio diverso, basato su ML, il riconoscimento di questi eventi. Invece, per incrementare ulteriormente, i risultati ottenuti in questo lavoro di tesi, per quegli eventi per i quali si hanno già valori di f-score elevati, si può pensare di arricchire il sistema di regole con ulteriori clausole o ulteriori regole logiche. Si pensi che il sistema di regole è per definizione estensibile nel numero di regole.

Altra forma di miglioramento è legata ad un incremento prestazionale, seppur non critico in questo dominio di lavoro. Le partite di gioco sono numerose per un dataset da 250 minuti (si faccia riferimento al *testing set*) e dal momento che non dipendono le une dalle altre, si può pensare di sottoporle al processamento di riconoscimento in maniera del tutto distinta e separata. Questa caratteristica dei dati consente di valutare alcune ipotesi circa l'utilizzo di un sistema multi-threading che, operando con flussi di esecuzione paralleli, può dapprima generare la lista di eventi complessi e, subito dopo, in cascata, eseguire l'operazione di valutazione. I vari flussi di esecuzione convergeranno poi tutti, dopo la fase di valutazione, e prima delle fase di presentazione dei risultati. La rappresentazione mediante grafico dei risultati ottenuti dovrà, infatti, contenere l'insieme globale degli esiti locali. C'è però da considerare che mentre il

processamento in parallelo ne riduce notevolmente il tempo totale impiegato ne aumenta il consumo di memoria dell'elaboratore, anche se il consumo di memoria non risulta eccessivo perché si pu fare uso dell'operazione di Garbage Collection inclusa tra le funzionalità di ETALIS.

```
Match_2019_02_11_#001
[XML_to_ETALIS]  Preprocessed data in: 0.5s.
[ETALIS]         Processed data in: 5.0s.
[ETALIS_to_XML]  Postprocessed data in: 1.5s.
[EVALUATOR]      Evaluated data in: 0.4s.

Match_2019_02_12_#001
[XML_to_ETALIS]  Preprocessed data in: 4.0s.
[ETALIS]         Processed data in: 88.2s.
[ETALIS_to_XML]  Postprocessed data in: 10.2s.
[EVALUATOR]      Evaluated data in: 3.0s.
```

Figura 34 - Esempio di prestazioni, mostrate nel terminale all'esecuzione del programma.

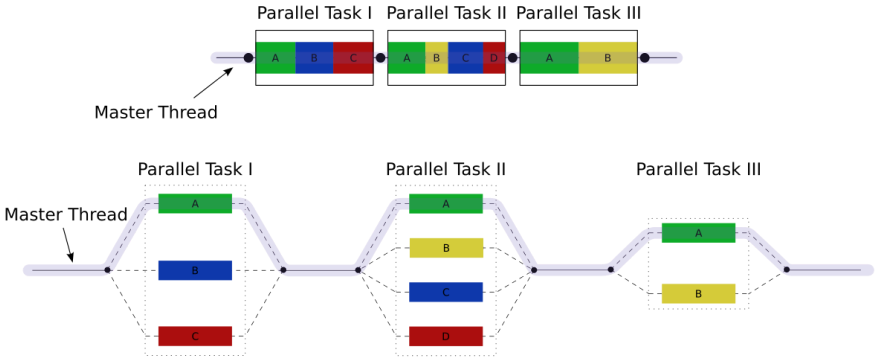


Figure 35 - Esempio di più thread che operano in parallelo e poi convergono nel flusso di esecuzione mediante join.

Bibliografia

[1] M. Davis, "Media Streams: an iconic visual language for video annotation," *Proceedings 1993 IEEE Symposium on Visual Languages*, Bergen, Norway, 1993, pp. 196-202.

[2] H. Miyamori and S. -. Iisaku, "Video annotation for content-based retrieval using human behavior analysis and domain knowledge," *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, Grenoble, France, 2000, pp. 320-325.

[3] Vondrick, C., Patterson, D. & Ramanan, D. Int J Comput Vis (2013) 101: 184.
https://doi.org/10.1007/s11263-012-0564-1

[4] M. Wang, X. Hua, R. Hong, J. Tang, G. Qi and Y. Song, "Unified Video Annotation via Multigraph Learning," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 5, pp. 733-746, May 2009.

[5] H. Shih, ”A Survey of Content-Aware Video Analysis for Sports”, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 28, No. 5, May 2011, pp. 1212-1231, DOI 10.1109/TCSVT.2017.2655624

[6] V. Tovinkere, R. J. Qian, ”Detecting semantic events in soccer games: Towards a complete solution”, Multimedia and Expo, September 2001, pp. 833 - 836, DOI 10.1109/ICME.2001.1237851

[7] Khan, Abdullah and Lazzerini, Beatrice and Calabrese, Gaetano and Serafini, Luciano, ”Soccer Event Detection”, 4th International Conference on Image Processing and Pattern Recognition (IPPR 2018), April 2018, pp. 119-129, DOI 10.5121/csit.2018.80509

[8] Shoham, Y. and Goyal, N., 1988. Temporal reasoning in artificial intelligence. In *Exploring artificial intelligence* (pp. 419-438). Morgan Kaufmann.

[9] Della Monica, D., Goranko, V., Montanari, A. and Sciavicco, G., 2013. Interval temporal logics: a journey. *Bulletin of EATCS*, *3*(105).

[10] Soccer Video and Player Position Dataset, http://home.ifi.uio.no/paalh/dataset/alfheim/

[11] S. Giancola, M. Amine, T. Dghaily and B. Ghanem, "SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos", 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, 2018, pp. 1792-179210, DOI: 10.1109/CVPRW.2018.00223

[12] Issia CNR, http://www.issia.cnr.it/wp/dataset-cnr-fig/

[13] Magglingen 2013, https://old.datahub.io/dataset/magglingen2013

[14] Multi-view Body Part Recognition with Random Forests
V. Kazemi, M. Burenius, H. Azizpour, J. Sullivan.
In Proceedings of BMVC 2013.

[15] Gameplay Football, properlydecent.com

[16] Keven Richly, Max Bothe, Tobias Rohloff, Christian Schwarz, ”Recognizing Compound Events in Spatio-Temporal Football Data”, International Conference on Internet of Things and Big Data, Apr 23 - 25, 2016, Rome, Italy DOI 10.5220/0005877600270035

[17] J. Lee and D. Nam and S. Moon and J. Lee and W. Yoo, ”Soccer event recognition technique based on pattern matching”, Federated Conference on Computer Science and Information Systems (FedCSIS), September 2017, pp. 643-646, DOI 10.15439/2017F104

[18] A. Gaidon and Z. Harchaoui and C. Schmid, ”Actom sequence models for efficient action detection”, CVPR, 2011, pp. 3201 - 3208, DOI 10.1109/CVPR.2011.5995646

[19] Peters, A., 2012. Design of a coaching support system for evaluating sport tactics.

[20] Kifer, M. and Subrahmanian, V.S., 1992. Theory of generalized annotated logic programming and its applications. *The Journal of Logic Programming*, *12*(4), pp.335-367.

[21] Kontchakov, R., Kurucz, A., Wolter, F. and Zakharyaschev, M., 2007. Spatial logic+ temporal logic=?. In *Handbook of spatial logics* (pp. 497-564). Springer, Dordrecht.