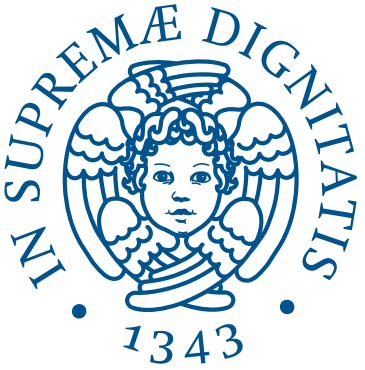


UNIVERSITÀ DI PISA

Computer Engineering

Formal Methods for Secure Systems

Relazione di Progetto

MEMBRI DEL TEAM:
Matteo Biondi
Olgerti Xhanej

Anno Accademico: 2020/2021

Contents

1 — Introduzione

1.1 Descrizione del problema

Tramite il software Into-CPS viene richiesto di modellare degli scenari con una following car che insegue una leading Car ad una distanza desiderata di 15m. L'unica dimensione presa in oggetto è l'asse x.

L'obiettivo del progetto è il seguente: analizzare possibili attacchi al suddetto sistema che possono causare uno scontro tra i due veicoli. In particolar modo implementare attacchi "Data Alteration" contro la posizione e l'accelerazione della following car.

Il presente documento viene redatto prendendo in esame solo il dominio delle nostre simulazioni. Di ogni affermazione sui risultati non è garantita la veridicità su un dominio più ampio.

2 — Scelte di Sviluppo

2.1 Strategia di Attacco

Gli attacchi verrano implementati utilizzando la tecnica del *Man-in-the-Middle*: verrà introdotta una FMU semplificata tra un punto di comunicazione di due FMU. Questo consentirà di semplificare la modifica dell'implementazione dell'attacco in quanto non è necessario conoscere i dettagli implementativi delle (altre) FMU in gioco. Questo comporta però un maggior overhead del sistema per effettuare la comunicazione dei parametri tra le varie FMU.

2.2 Scelta dei parametri

- **Step-size**: **0.01s**. E' un buon trade-off tra un sensoring più preciso ed una durata di simulazione accettabile.
- **Tempo di Simulazione**: **100s**. Abbiamo valutato questo tempo come un ragionevole trade-off tra la capacità di computazione delle nostre macchine ed i risultati che possiamo mettere in luce.

3 — Implementazione

3.1 VanillaCase

Nella seguente figura è possibile osservare le connessioni logiche tra tre FMU principali:

- **FMU of the leading car**: questa FMU implementa il comportamento della leading car. Per funzionare non ha bisogno di alcun input da altre FMU e produce in output la posizione della macchina, la velocità e la sua accelerazione.
- **FMU of the following algorithm**: questa FMU implementa l'algoritmo di inseguimento. Presi in ingresso i parametri di posizione, velocità e accelerazione della leading car ed i parametri di posizione e velocità della following car produce in output l'accelerazione per la following car.
- **FMU of the following car**: questa FMU implementa il comportamento della following car. Per funzionare prende in ingresso l'accelerazione dalla precedente FMU e produce in output la sua posizione e velocità.

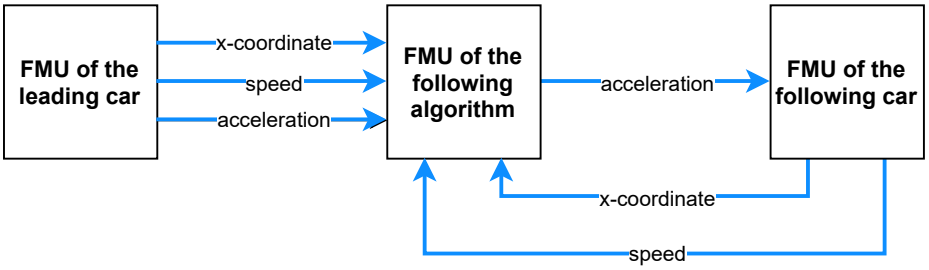


Figure 1: Multi-Model schema del VanillaCase

In figura 2 viene rappresentata l'overview del relativo Multi-Model sviluppato con il tool INTO-CPS.

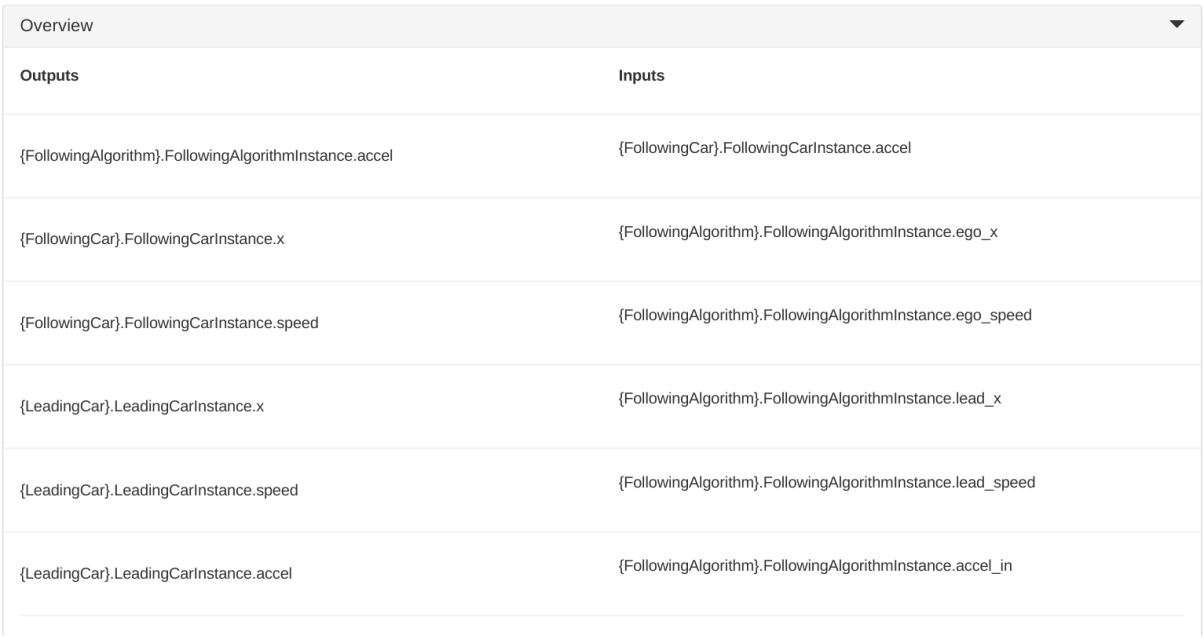


Figure 2: Multi-Model Overview del Vanilla Case

3.2 Attacco all'accelerazione

A differenza dello schema presentato nel VanillaCase, viene ora aggiunta una ulteriore FMU situata fra "FMU of the following algorithm" e "FMU of the following car" già presenti. La nuova FMU implementa con strategia *Man-in-the-Middle* un attacco di tipo data alteration sull'accelerazione passata tra il Following Algorithm e la Following Car. Fare riferimento alla sezione 3.5 per dettagli sul comportamento dell'attacco.

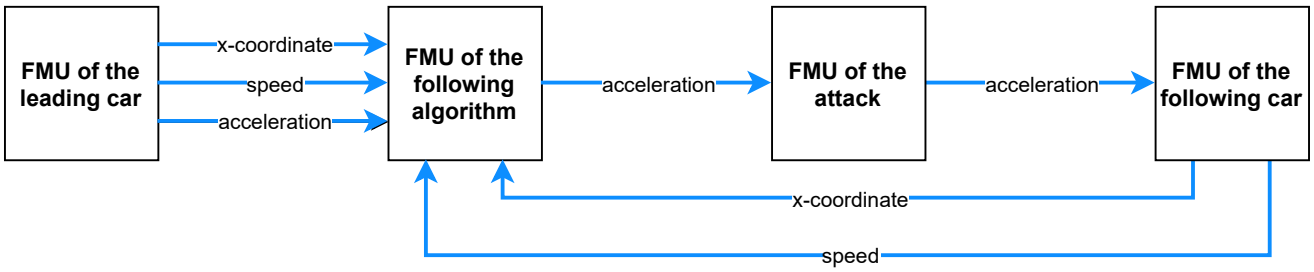
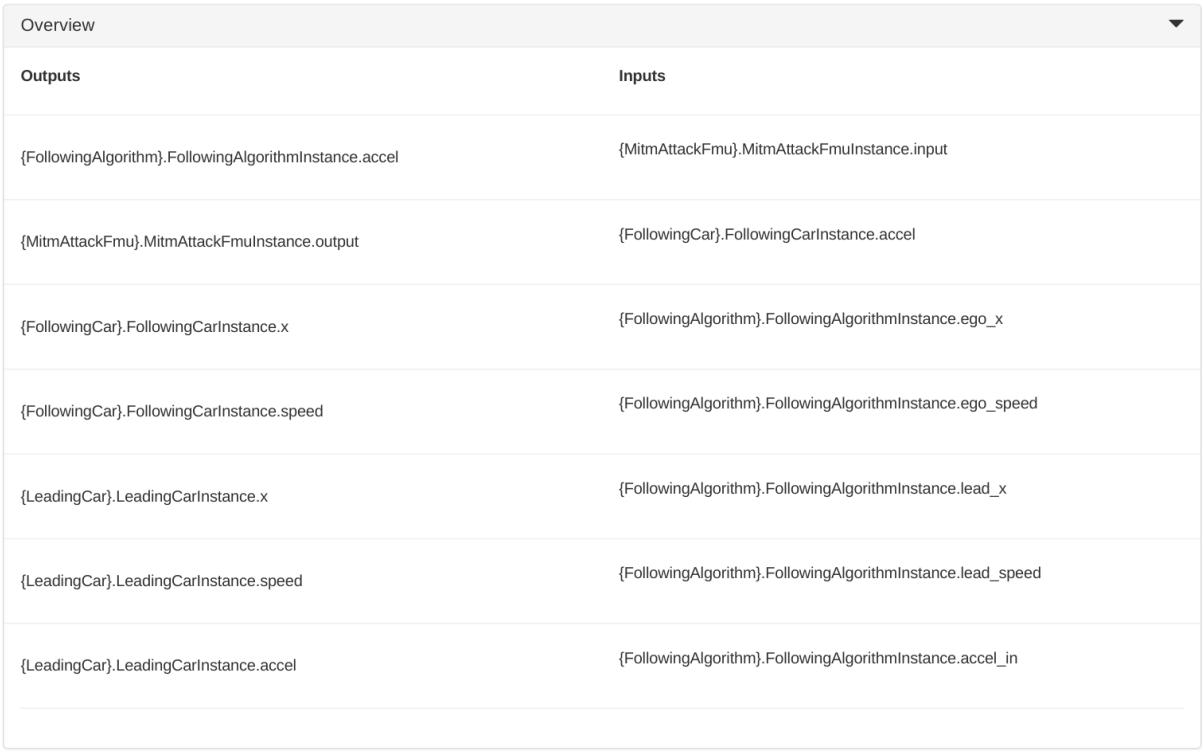


Figure 3: Multi-Model schema dell'Attacco alla Accelerazione

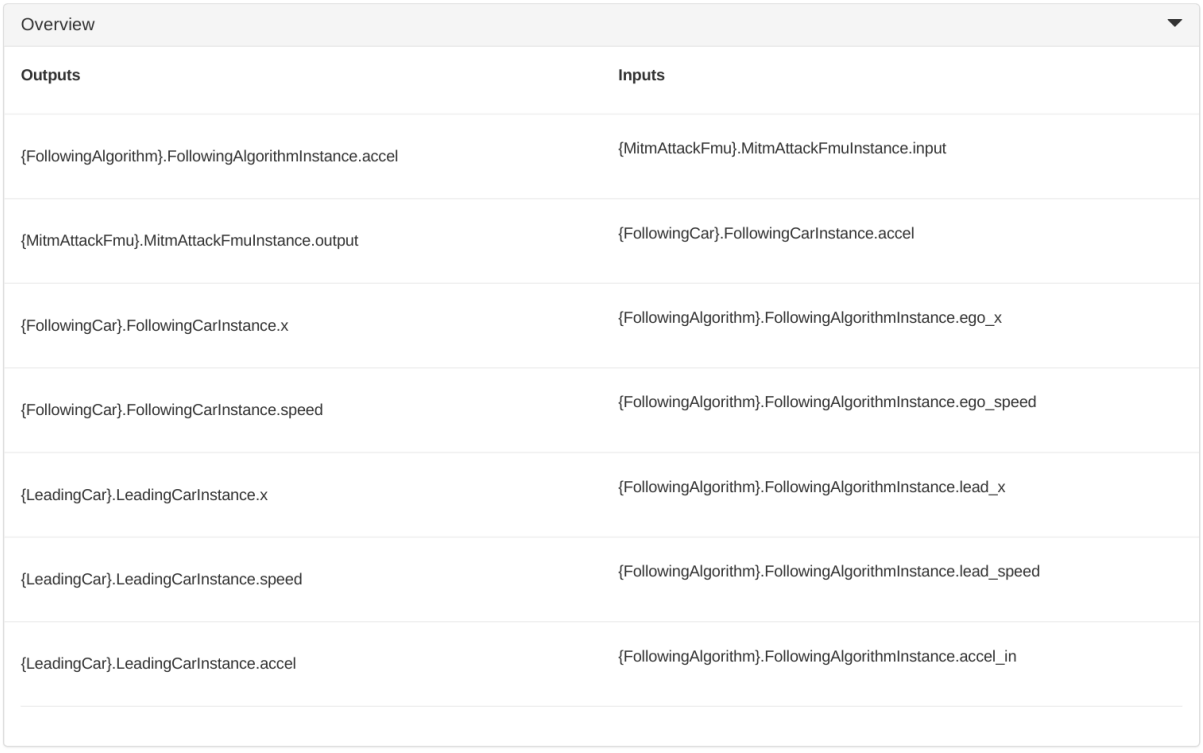
In figura 4 e 5 viene rappresentata l'overview del relativo Multi-Model sviluppato con il tool INTO-CPS.



Overview

Outputs	Inputs
{FollowingAlgorithm}.FollowingAlgorithmInstance.accel	{MitmAttackFmu}.MitmAttackFmuInstance.input
{MitmAttackFmu}.MitmAttackFmuInstance.output	{FollowingCar}.FollowingCarInstance.accel
{FollowingCar}.FollowingCarInstance.x	{FollowingAlgorithm}.FollowingAlgorithmInstance.ego_x
{FollowingCar}.FollowingCarInstance.speed	{FollowingAlgorithm}.FollowingAlgorithmInstance.ego_speed
{LeadingCar}.LeadingCarInstance.x	{FollowingAlgorithm}.FollowingAlgorithmInstance.lead_x
{LeadingCar}.LeadingCarInstance.speed	{FollowingAlgorithm}.FollowingAlgorithmInstance.lead_speed
{LeadingCar}.LeadingCarInstance.accel	{FollowingAlgorithm}.FollowingAlgorithmInstance.accel_in

Figure 4: Multi-Model Overview dell'attacco all'accelerazione (caso attacco semplice)



Overview

Outputs	Inputs
{FollowingAlgorithm}.FollowingAlgorithmInstance.accel	{MitmAttackFmu}.MitmAttackFmuInstance.input
{MitmAttackFmu}.MitmAttackFmuInstance.output	{FollowingCar}.FollowingCarInstance.accel
{FollowingCar}.FollowingCarInstance.x	{FollowingAlgorithm}.FollowingAlgorithmInstance.ego_x
{FollowingCar}.FollowingCarInstance.speed	{FollowingAlgorithm}.FollowingAlgorithmInstance.ego_speed
{LeadingCar}.LeadingCarInstance.x	{FollowingAlgorithm}.FollowingAlgorithmInstance.lead_x
{LeadingCar}.LeadingCarInstance.speed	{FollowingAlgorithm}.FollowingAlgorithmInstance.lead_speed
{LeadingCar}.LeadingCarInstance.accel	{FollowingAlgorithm}.FollowingAlgorithmInstance.accel_in

Figure 5: Multi-Model Overview dell'attacco all'accelerazione (caso attacco multiplo)

3.3 Attacco alla Posizione

A differenza dello schema presentato nel VanillaCase, viene ora aggiunta una ulteriore FMU situata fra "FMU of the following car" e "FMU of the following algorithm" già presenti. La nuova FMU implementa con strategia *Man-in-the-Middle* un attacco di tipo data alteration sulla posizione passata tra la Following Car e il Following Algorithm. Fare riferimento alla sezione 3.5 per dettagli sul comportamento dell'attacco.

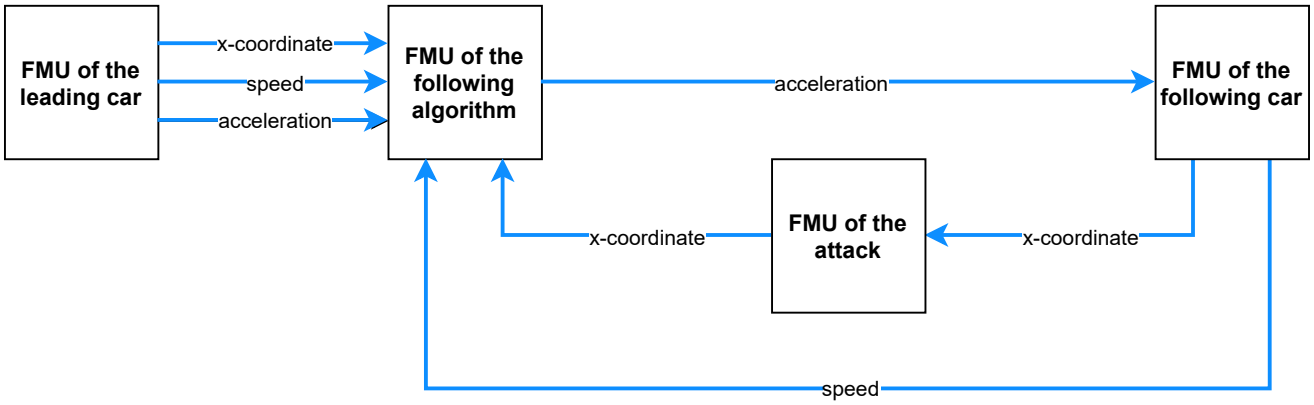


Figure 6: Multi-Model schema dell'Attacco alla Posizione

In figura 7 e 8 viene rappresentata l'overview del relativo Multi-Model sviluppato con il tool INTO-CPS.

Overview

Outputs	Inputs
{FollowingAlgorithm}.FollowingAlgorithmInstance.accel	{FollowingCar}.FollowingCarInstance.accel
{FollowingCar}.FollowingCarInstance.x	{Attack}.AttackInstance.input
{FollowingCar}.FollowingCarInstance.speed	{FollowingAlgorithm}.FollowingAlgorithmInstance.ego_speed
{Attack}.AttackInstance.output	{FollowingAlgorithm}.FollowingAlgorithmInstance.ego_x
{LeadingCar}.LeadingCarInstance.x	{FollowingAlgorithm}.FollowingAlgorithmInstance.lead_x
{LeadingCar}.LeadingCarInstance.speed	{FollowingAlgorithm}.FollowingAlgorithmInstance.lead_speed
{LeadingCar}.LeadingCarInstance.accel	{FollowingAlgorithm}.FollowingAlgorithmInstance.accel_in

Figure 7: Multi-Model Overview dell'attacco alla posizione (caso attacco semplice)

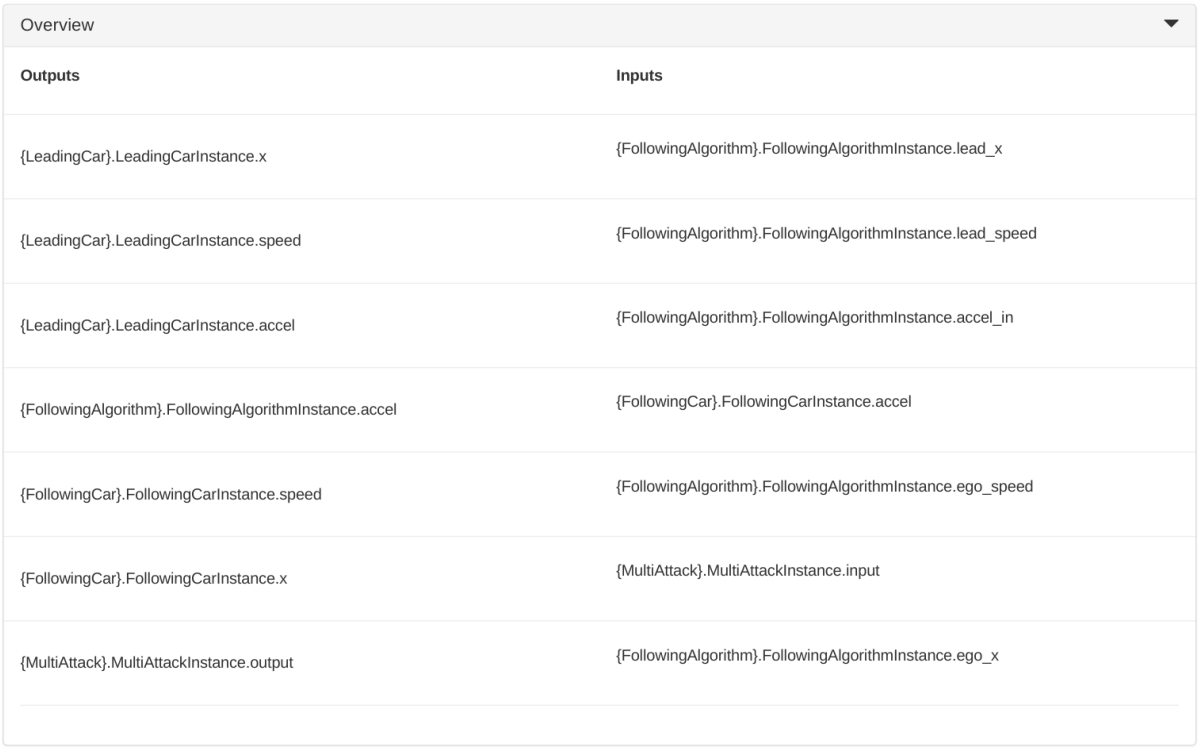


Figure 8: Multi-Model Overview dell'attacco alla posizione (caso attacco multiplo)

3.4 Configurazione in Comune

La configurazione delle seguenti FMU verrà applicata per tutte le simulazioni che verranno effettuate.

- **LeadingCar**:
  - Posizione iniziale **x0**: 50m
  - Velocità iniziale **v0**: 0m/s
- **FollowingAlgorithm**:
  - **c1**: 0.5
  - **eps**: 1
  - **omega_n**: 0.2
- **FollowingCar**:
  - Posizione iniziale **x0**: 0m
  - Velocità iniziale **v0**: 0m/s

3.5 Comportamento degli Attacchi

L'FMU che verrà utilizzata negli attacchi MITM presenterà due implementazioni diverse:

- **Attacco Semplice**: l'attacco consiste nel modificare l'input dell'FMU di attacco con il valore del parametro **attack_value** dall'istante temporale **attack_time** fino al termine della simulazione. Tale valore viene restituito in output dall'FMU di attacco. Tale FMU è implementata tramite il file Attack_fmu.fmu. Sono pertanto riportati due estratti di codice implementativo della suddetta FMU: nel primo estratto vi è rappresentata la struttura che rappresenta lo stato dell'fmu. Nel secondo caso è rappresentata l'evoluzione dello stato ad ogni step_size.

```
//in sources/misraC/Attack_fmu.h
typedef struct {
  Mode mode;
  Mode previous_mode;
  float64_t attack_time; //-- real
  float64_t attack_value; //-- real
  float64_t input; //-- real
  float64_t output; //-- real
  float64_t step_size; //-- real
  float64_t time; //-- real
} State;
```

```
//in misraC/Attack_fmu.c
State* tick(State* st) {
  // assert( per_tick(st) );
  if (st->mode == X1 && ( st->time < st->attack_time )) {
    #ifdef DBG
    _dbg_print_condition("st->mode == X1 && ( st->time < st->
 attack_time )");
    #endif
    leave(X1, st);
    st->output = st->input;
    st->time = st->time + st->step_size;
    enter(X1, st);
  } else if (st->mode == X1 && ( st->time >= st->attack_time )) {
    #ifdef DBG
    _dbg_print_condition("st->mode == X1 && ( st->time >= st->
 attack_time )");
    #endif
    leave(X1, st);
    st->output = st->attack_value;
    st->time = st->time + st->step_size;
    enter(X1, st);
  }
  #ifdef DBG
  _dbg_print_state(st);
  #endif
  return st;
}

```

- **Attacco Multi-step**: l'attacco consiste nel modificare l'input dell'FMU di attacco con il valore del parametro **attack_value** per un tempo pari a **attack_duration**, ripetuto **attack_occurrencies** volte e separato nel tempo da **attack_distance** secondi. Tale valore viene restituito in output dall'FMU di attacco. L'attacco inizierà dall'istante temporale **attack_time**. Tale FMU è implementata tramite il file MultiStep_MultiAttacks_Fmu.fmu. Sono mostrati pertanto due estratti della relativa documentazione: nel primo estratto viene rappresentata la differenza in

termini di stato dell'Fmu rispetto al caso precedente. Nel secondo estratto viene rappresentata l'evoluzione dello stato dell'Fmu.

```

//in sources/misraC/Attack_fmu.h
//Sono sottolineate le modifiche alla struttura dello stato
typedef struct {
  ...
  float64_t attack_duration; //-- real
  uint64_t attack_occurrences; //-- integer
  float64_t attack_distance; //-- real
} State;

```

```
//in misraC/Attack_fmu.c
//Sono sottolineate le modifiche alla tick function
bool under_attack(State* st){
  float64_t attack_time_i;
  if(st->time >= st->attack_time) {
    for (unsigned int i = 0; i < st->attack_occurrences; ++i){
      //individuo l'inizio di ogni intervallo di attacco
      attack_time_i = st->attack_time + (st->attack_duration + st->attack_distance)*i;
      //vedo se sono nel mezzo di un intervallo
      if(st->time >= attack_time_i && st->time <= attack_time_i + st->attack_duration)
      return true;
    }
  }
  return false;
}

State* tick(State* st) {
  // assert( per_tick(st) );
  if (st->mode == X1 && !under_attack(st)) {
    #ifdef DBG
    _dbg_print_condition("st->mode == X1 && ( st->time < st->attack_time )");
    #endif
    leave(X1, st);
    st->output = st->input;
    st->time = st->time + st->step_size;
    enter(X1, st);
  } else if (st->mode == X1 && under_attack(st)) {
    #ifdef DBG
    _dbg_print_condition("st->mode == X1 && ( st->time >= st->attack_time )");
    #endif
    leave(X1, st);
    st->output = st->attack_value;
    st->time = st->time + st->step_size;
    enter(X1, st);
  }
  #ifdef DBG
  _dbg_print_state(st);
  #endif
  return st;
}
```

4 — Analisi dei Risultati

4.1 VanillaCase

4.1.1 Risultati Co-Simulazione

E' stata effettuata una simulazione nel caso base per accertarsi che il comportamento del sistema conduca alla convergenza delle due macchine.

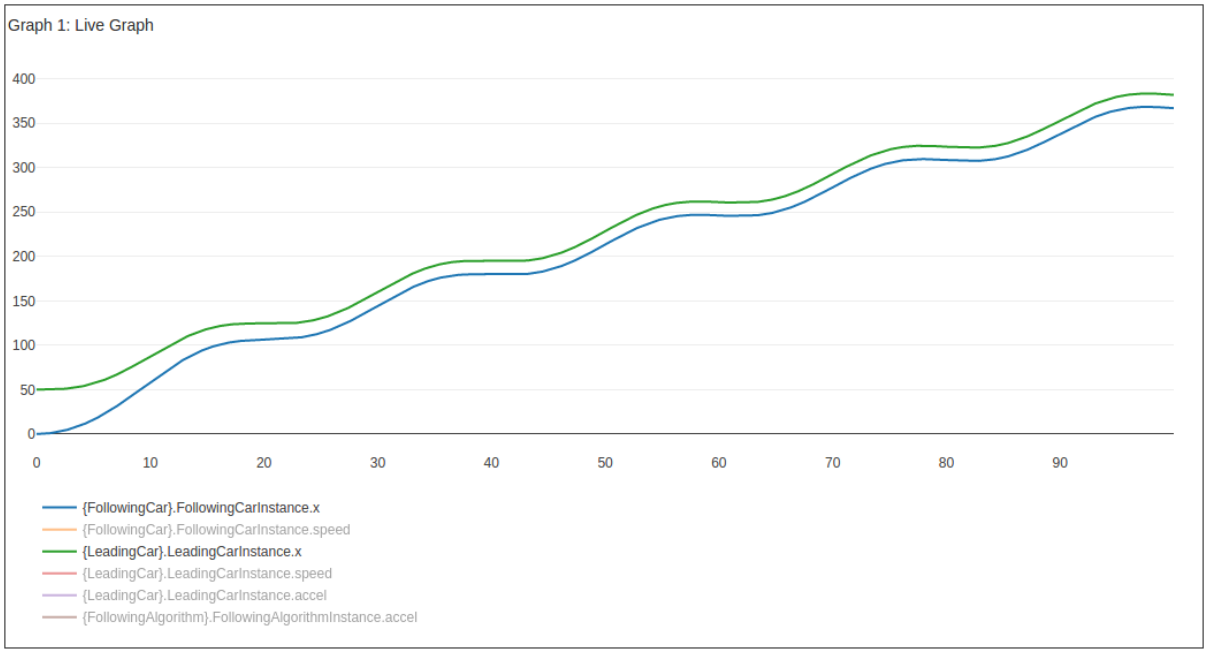


Figure 9: Posizione x della LeadingCar (verde) e FollowingCar (blu)

La distanze media tra le due auto è pari a **18.49m**. Dopo un iniziale periodo di transizione di circa 20s il sistema raggiunge la convergenza attesa e i due veicoli proseguono il percorso ad una distanza approssimativa di 15m fino a fine simulazione.

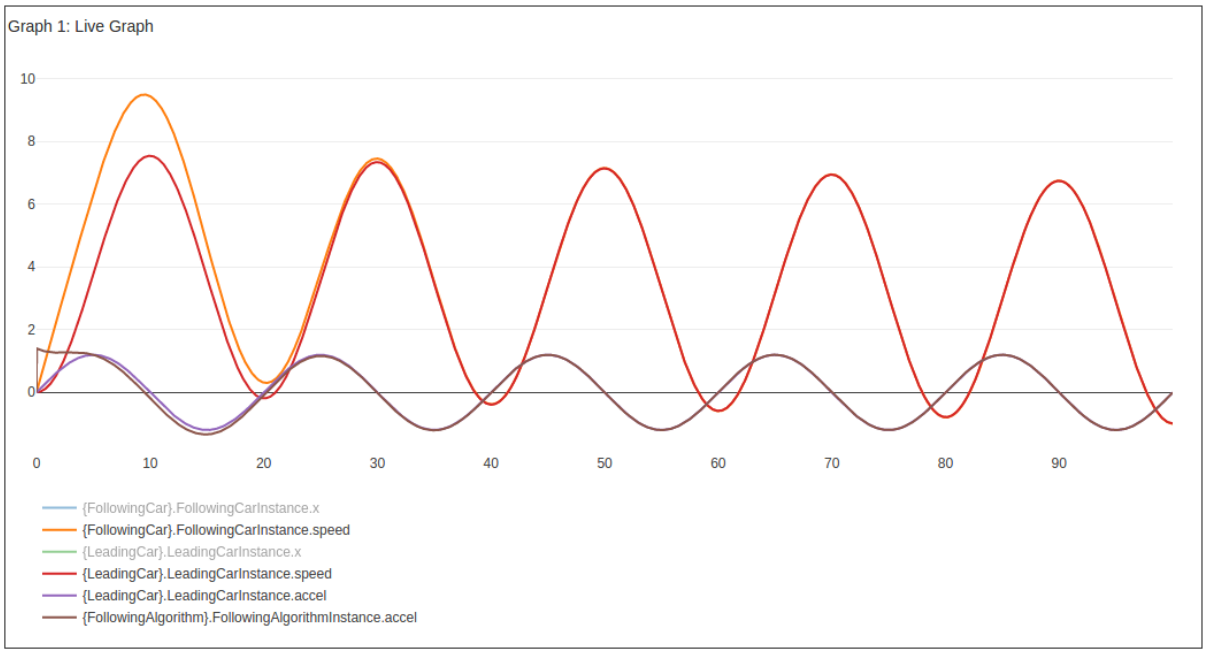


Figure 10: Velocità e accelerazione della LeadingCar (rispettivamente rossa e viola) e FollowingCar (rispettivamente gialla e marrone)

Dalla figura sopra riportata è inoltre osservabile come negli istanti iniziali la Following Car abbia una accelerazione positiva maggiore di quella della Leading Car. Questo si riflette inoltre sulle relative velocità. Il motivo di questo comportamento è dovuto all'iniziale periodo di transizione in cui la Following Car recupera la distanza iniziale (molto maggiore di 15m) dalla Leading Car.

4.2 Attacco all'accelerazione

4.2.1 Attacco Semplice

Risultati DSE Come primo approccio all'analisi al sistema è stato scelto di fare uso del DSE, configurato andando a variare l'**attack_value** e l'**attack_time** con i seguenti parametri:

- **Attack_value**: [-5, -1, 0, 1, 5] m/s^2
- **Attack_time**: [0s, .., 40s] con step a 5

I risultati ottenuti sono stati successivamente eleborati così da estrapolare il seguente grafico che mostra la percentuale degli incidenti per ogni **attack_value** al variare di **attack_time**. Per individuare le condizioni di attacco è stato necessario estrapolare la distanza minima delle due macchine sull'intero tempo di simulazione.

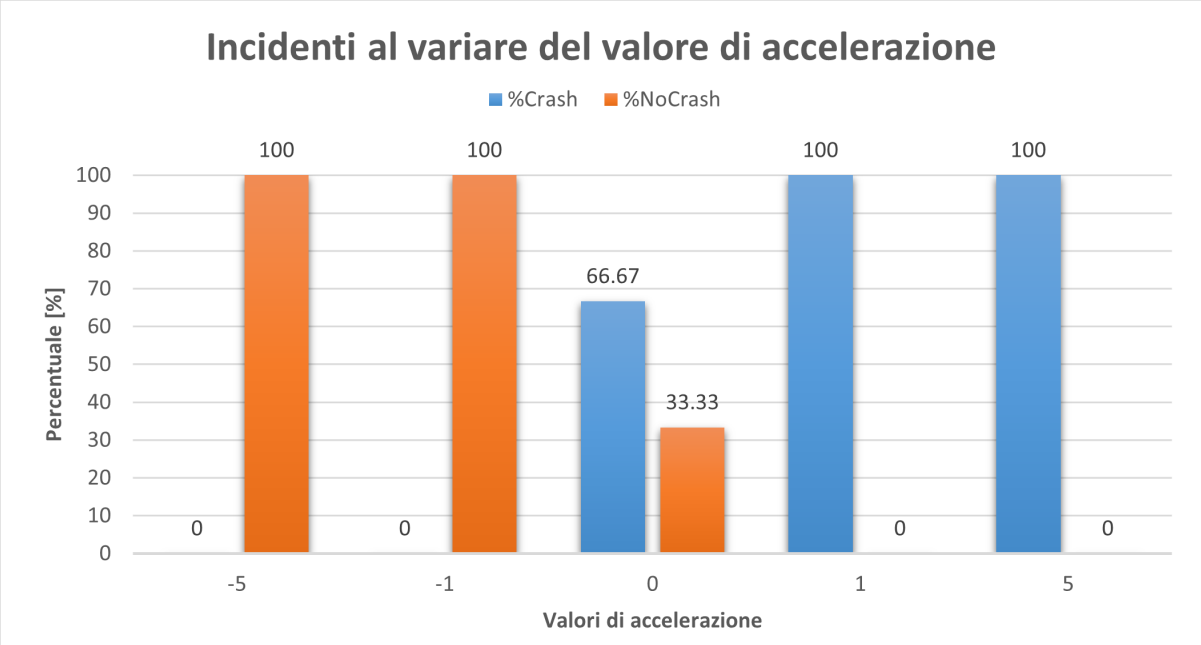


Figure 11: Rappresentazione delle percentuali di incidenti nei casi testati con studio DSE

Come si può notare, è possibile individuare tre casi ben distinti:

- **Attacchi con accelerazione negativa**: La Following Car è portata a rallentare con andamento lineare fino a cambiare la propria direzione di marcia. In questo caso le macchine tendono ad allontanarsi e l'incidente non avrà luogo nel dominio dell'esperimento. Inoltre è doveroso sottolineare che la Following Car perde completamente la capacità di inseguimento della Leading Car. Non ci sarà quindi convergenza fra le due vetture.

- **Attacchi con accelerazione pari a 0** m/s^2: Dal grafico emerge una chiara necessità di uno studio più approfondito di questa casistica in quanto non si delinea alcun risultato conclusivo. Essendo che l'accelerazione resta costante e pari a 0 m/s^2, la velocità della Following Car rimane costante al valore nel momento **Attack_time**. La presenza o meno di incidenti dipende quindi proprio dal valore della velocità e quindi da **Attack_time**

- **Attacchi con accelerazione positiva**: La Following Car è portata ad aumentare la propria velocità con andamento lineare. In questo caso le macchine tendono ad avvicinarsi e l'incidente avrà luogo.

Esistono tuttavia condizioni speciali che è doveroso sottolineare:

- **Attacchi con accelerazione negativa**: Se la Leading Car decelerasse con continuità (per un intervallo di tempo sufficientemente ampio) più di quanto non faccia la Following Car sotto attacco, allora in tal caso l'incidente avverrebbe.

- **Attacchi con accelerazione positiva**: Se la Leading Car accelerasse con continuità (per un intervallo di tempo sufficientemente ampio) più di quanto non faccia la Following Car sotto attacco, allora in tal caso l'incidente non avverrebbe.

Risultati Co-Simulazione Con l'obiettivo di rafforzare quanto appena descritto e individuato tramite l'analisi dei risultati del DSE, vengono qui riportati tre casi fondamentali.

Attacchi con accelerazione positiva pari a 1 m/s^2 Di seguito sono riportati i grafici in cui sono raffigurati le posizioni (Fig. 12), le velocità (Fig. 13) le accelerazioni (Fig. 14) delle macchine e l'accelerazione dell'algoritmo di inseguimento (Fig. 15). L'attacco è stato eseguito con:

- **attack_value**: 1 m/s^2
- **attack_time**: 20s

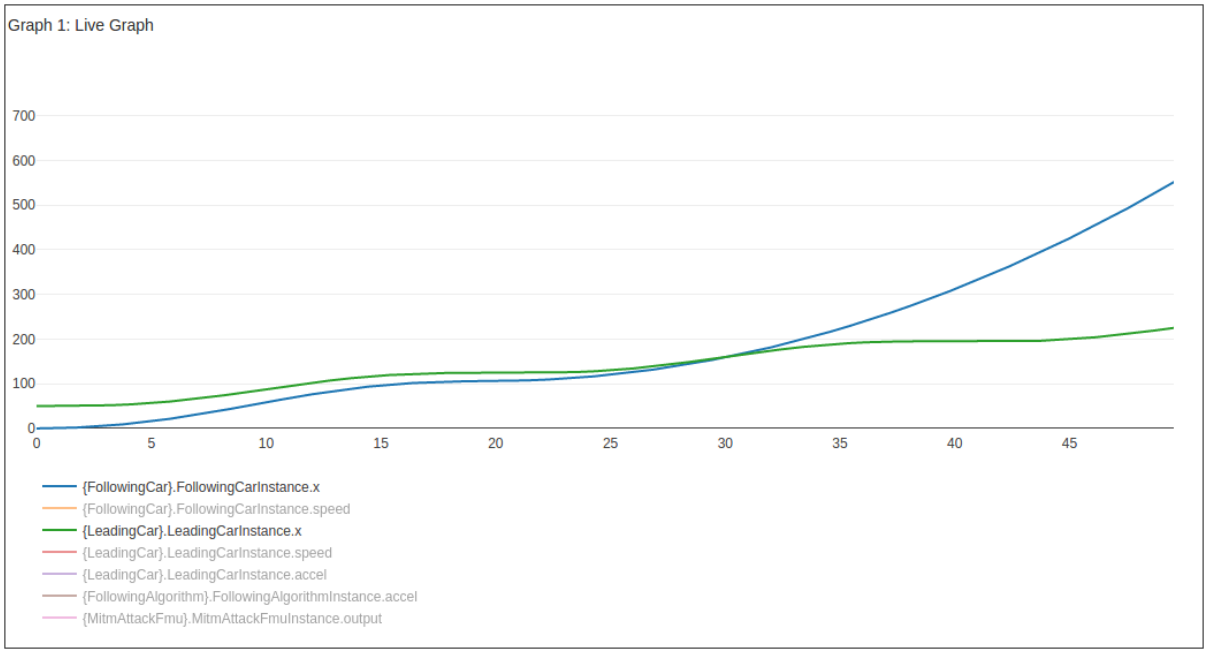


Figure 12: Ingrandimento del grafico delle posizioni dei due veicoli in questo caso di attacco semplice

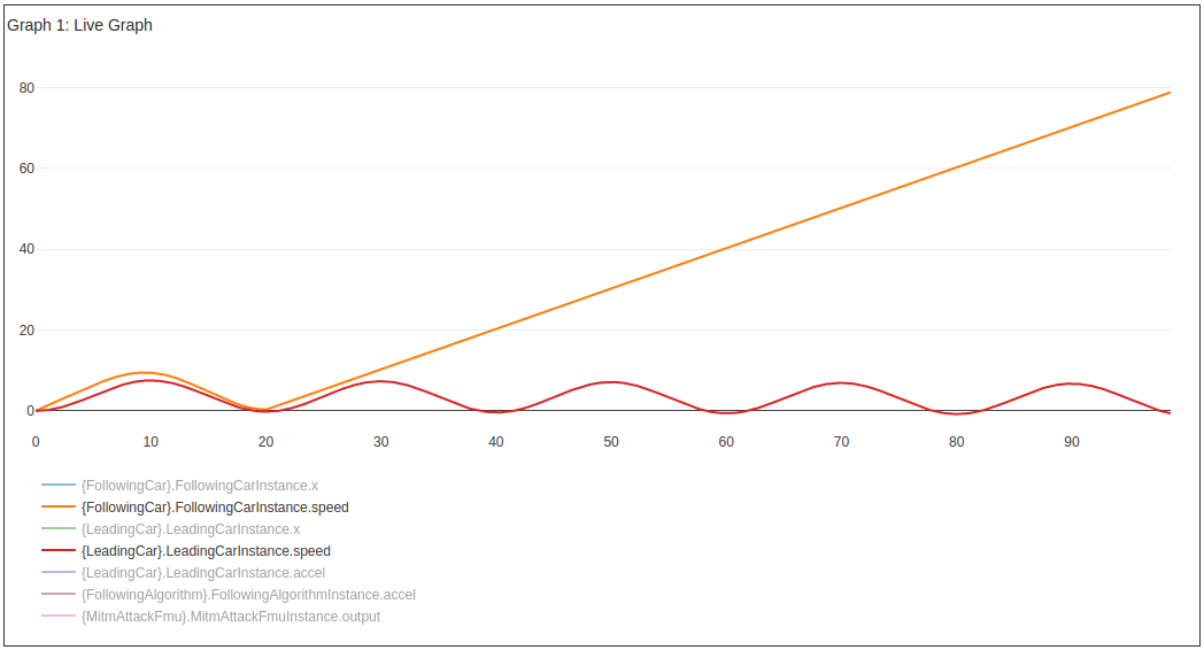


Figure 13: Grafico raffigurante le velocità dei due veicoli in questo caso di attacco semplice

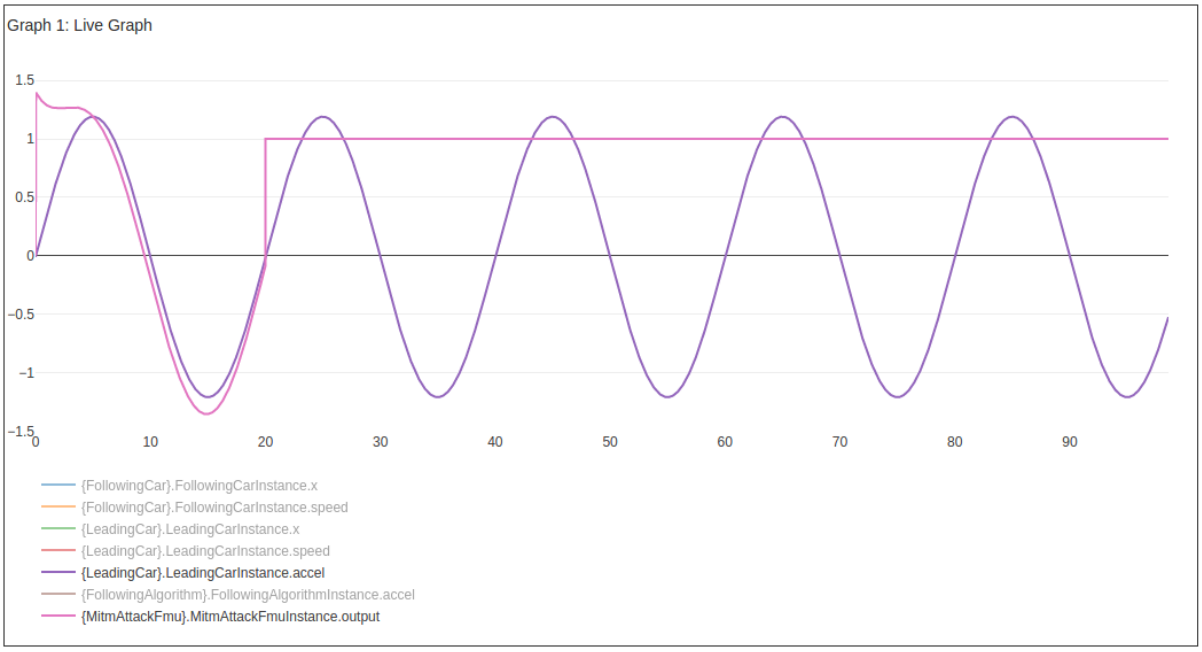


Figure 14: Grafico raffigurante le accelerazioni dei due veicoli in questo caso di attacco semplice

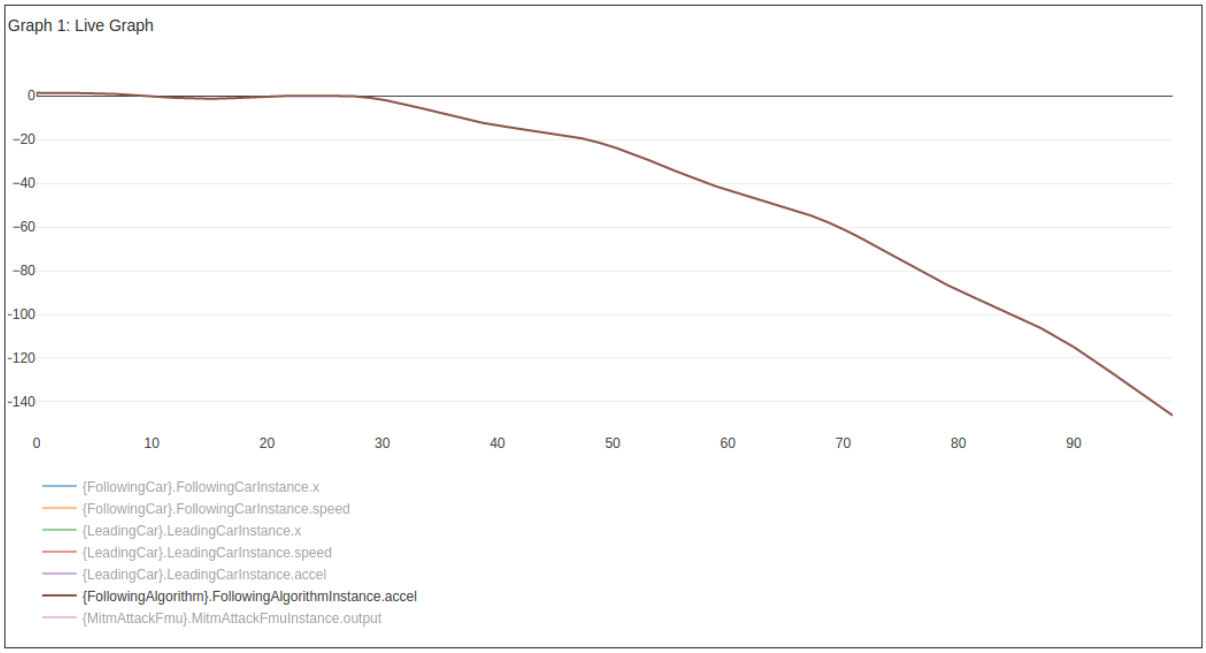


Figure 15: Grafico raffigurante l'accelerazione suggerita da Following Algorithm in questo caso di attacco semplice

Dalle osservazioni fatte si può evincere quanto segue:

- La Following Car e la Leading Car fanno un incidente. Essendo che l'accelerazione è costante e tale che $|Attack\_value| > 0$, allora la velocità tende ad aumentare linearmente. L'allontanamento da Leading Car avverrà in modo quadratico nel tempo
- L'accellerazione che Following Algorithm pensa di dire a Following Car è sempre minore con andamento non lineare. Avrà sicuramente delle micro-oscillazioni ma sono quasi impercettibili a causa dell'elevata distanza dalla Leading Car. Quindi una decellerazione/accellerazione della Leading Car ha un effetto quasi trascurabile su Following Algorithm

Attacchi con accelerazione negativa pari a -1 m/s^2 Di seguito viene riportato il grafico raffigurate le posizioni dei due veicoli (Fig. 16). I grafici di velocità e accelerazione sono deducibili dal lettore osservando quelli dell'attacco precedente. L'attacco è stato eseguito con:

- **attack_value**: -1 m/s^2
- **attack_time**: 20s

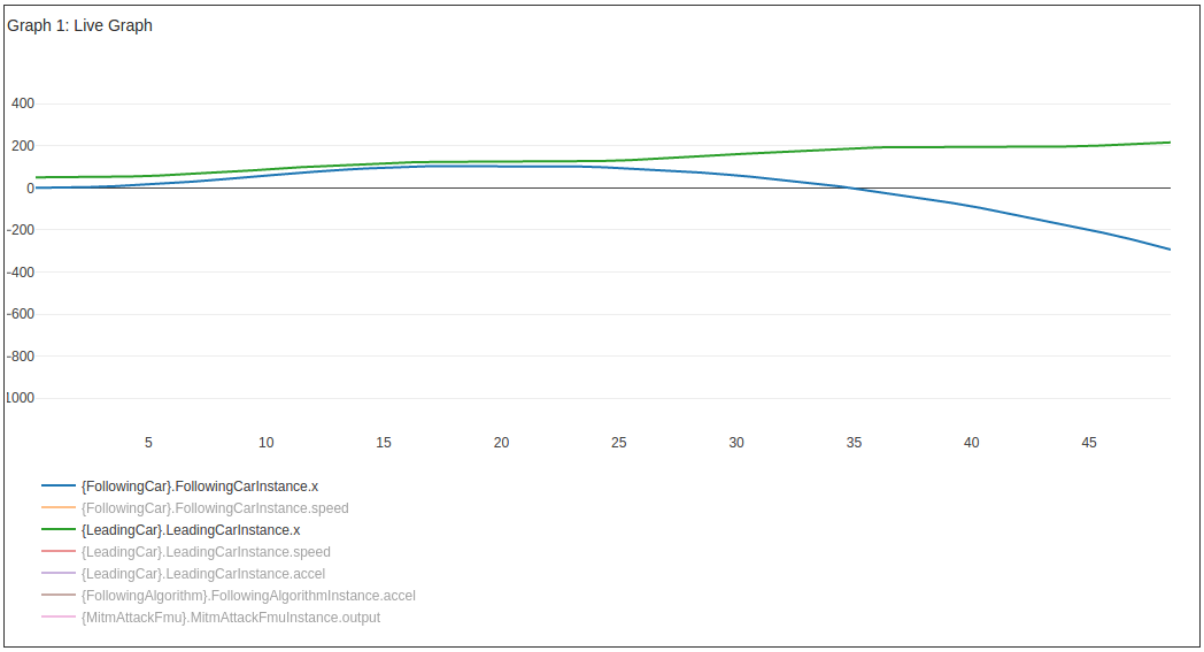


Figure 16: Ingrandimento del grafico delle posizioni dei due veicoli in questo caso di attacco semplice

Dalle osservazioni fatte si può evincere quanto segue:

- La Following Car non fa un incidente (nel dominio di simulazione) e continua la sua corsa in senso opposto rispetto alla Leading Car. Ogni considerazione fatta per il caso precedente rispetto a accelerazione e velocità sono ancora valide ma speculari.
- La velocità di Following Car decresce linearmente fino ad annullarsi e poi a cambiare segno (facendo muovere la macchina in retromarcia)
- Ogni considerazione fatta nel caso precedente rispetto all'accelerazione che Following Algorithm pensa di dire a Following Car è tutt'ora valida e speculare al caso precedente.

Attacchi con accelerazione pari a 0 m/s^2 Nella tabella seguente sono riportati le diverse simulazioni effettuate per il caso **attack_value = 0** m/s^2. Per ogni casistica viene indicato il diverso tempo di attacco, una idea approssimativa della velocità a cui la Following Car si stabilizza ed infine una breve descrizione di alcune osservazioni fatte sul risultato ottenuto (considerando il nostro dominio di simulazione).

Stato Convergenza	Tempo di Attacco [s]	Valore Velocità Following Car dopo Attacco	Risultato
Prima della Convergenza	10	Circa Valore Massimo	La following car fa un incidente. Accelerazione risultato di Following Algorithm ha andamento sinusoidale decrescente, posizione leading car non trascurabile
	15	Circa Valore Medio	Si verifica un incidente tra i veicoli. Accelerazione di Following Algorithm decrescente con andamento sinusoidale
	20	Circa Valore Minino	Following car non fa un incidente e continua la sua corsa distanziandosi dalla leading car. L'accelerazione risultato di Following Algorithm per following car ha un andamento sinusoidale e crescente
Dopo la Convergenza	40	Circa Valore Minimo	Accelerazione risultato di Following Algorithm crescente con andamento sinusoidale. Nessun incidente ma allontanamento con movimento di Following Car in senso opposto.
	45	Circa Valore Medio	Susseguirsi di avvicinamenti e allontanamenti fra i due veicoli. Se progredita nel tempo può portare ad un lento avvicinamento e ad incidente. Accelerazione di Following Algorithm ha un andamento sinusoidale decrescente.
	50	Circa Valore Massimo	Following Car fa un incidente con leading car. Accelerazione di Following Algorithm sinusoidale decrescente

A dimostrazione dell'elevata variabilità del risultato vengono ora proposti tre grafici raffiguranti le posizioni dei due veicoli nei tre casi a convergenza sopra descritti.

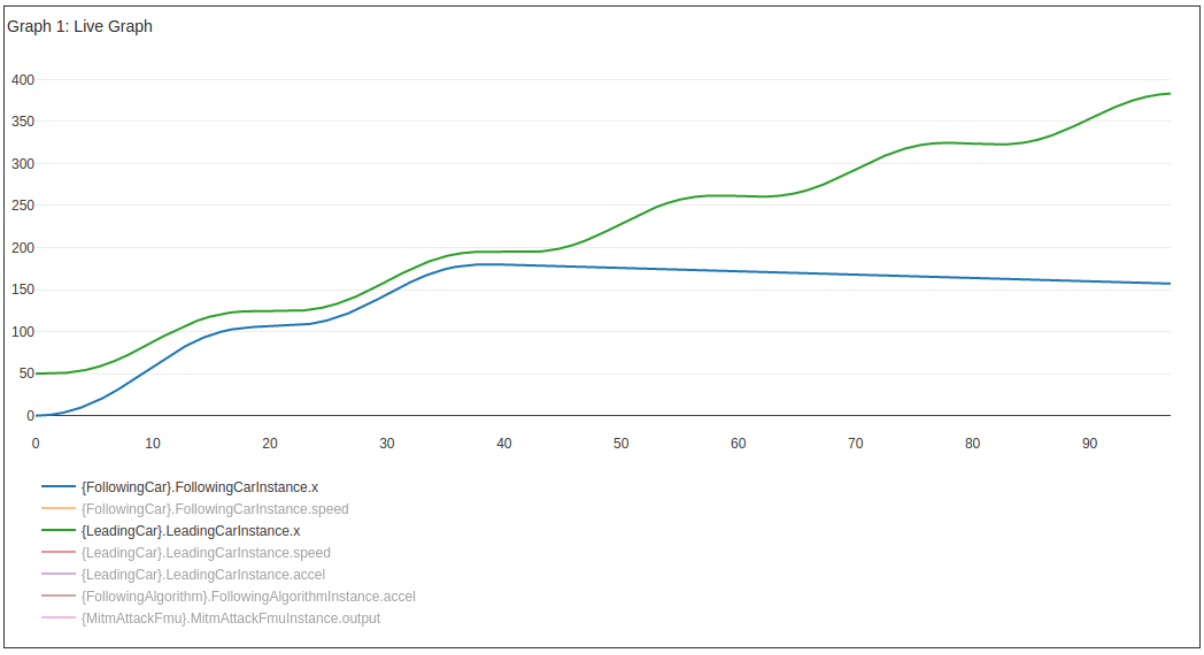


Figure 17: Grafico della posizione dei veicoli nel caso Tempo di Attacco a 40s

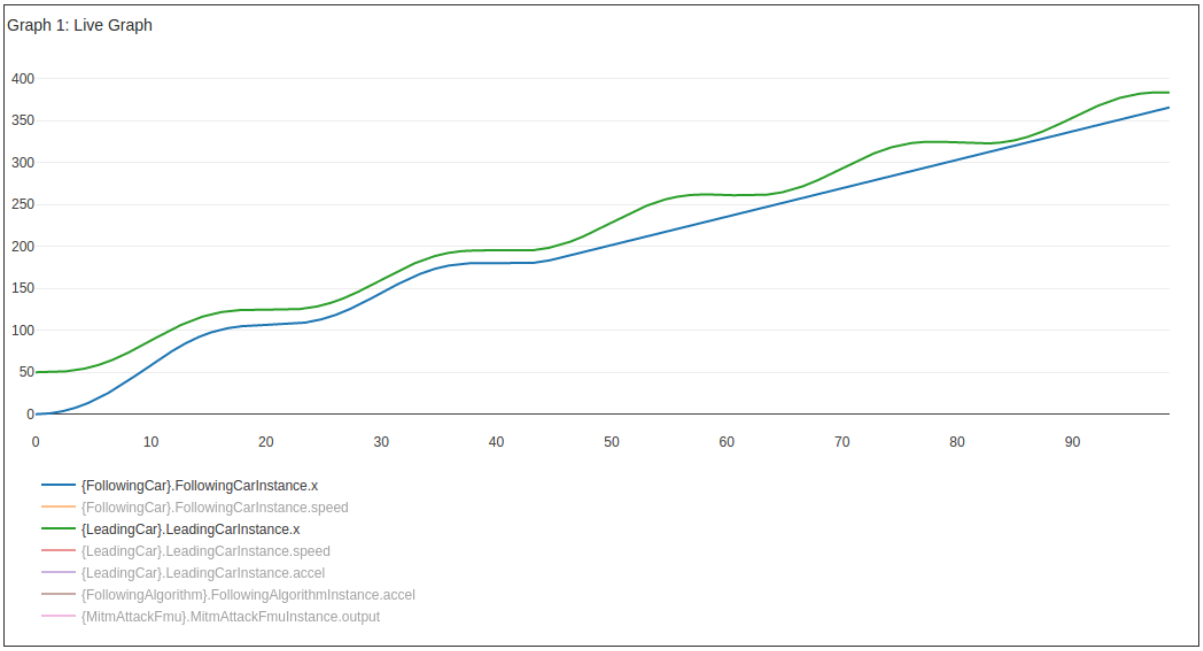


Figure 18: Grafico della posizione dei veicoli nel caso Tempo di Attacco a 45s

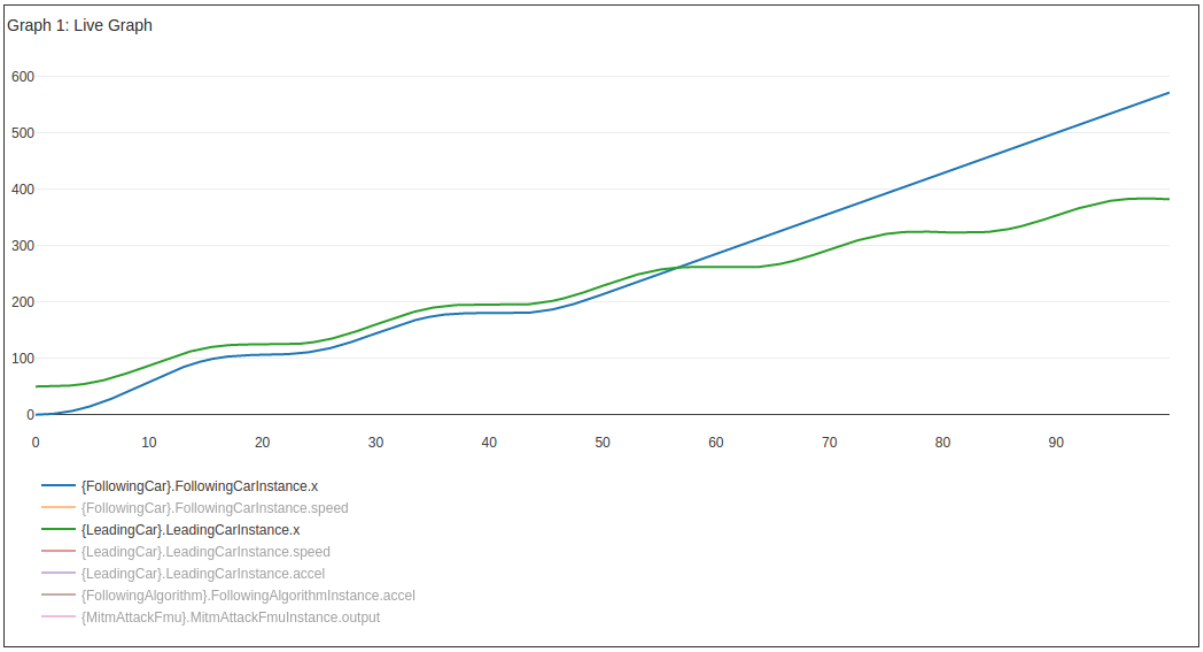


Figure 19: Grafico della posizione dei veicoli nel caso Tempo di Attacco a 50s

I grafici in Fig. 17 e 18, ad esempio, mostrano come, nonostante vengano simulati attacchi con **attack_time** molto simili, il risultato sia completamente diverso (nel dominio dell'esperimento).

4.2.2 Attacco Multiplo

In questa sezione vengono riportate due diverse condizioni di attacco in cui quest'ultimo ha una durata di un certo numero di step e si ripete più volte nel tempo. L'obiettivo

è quello di individuare una condizione in cui, nonostante gli attacchi ripetuti, il sistema risulta tollerante e uno invece in cui l'attacco porta ad un incidente fra i due veicoli

Risultati Co-Simulazione

Attacco senza incidente L'obiettivo della presente co-simulazione è quello di andare ad individuare un attacco in cui la presenza di più occorrenze risulta essere un elemento non chiave nel verificarsi di un incidente fra i due veicoli. In particolare viene posto come obiettivo quello di studiare il comportamento della Following Car al termine dell'attacco multiplo. Di seguito sono riportate le configurazioni dell'attacco in esame.

- **Attack_occurrencies**: 2
- **Attack_duration**: 5s
- **Attack_time**: 30s
- **Attack_value**: -5 m/s^2
- **Attack_distance**: 10s
- **Step_size**: 0.01s

Vengono ora riportati i risultati della co-simulazione nelle immagini seguenti.

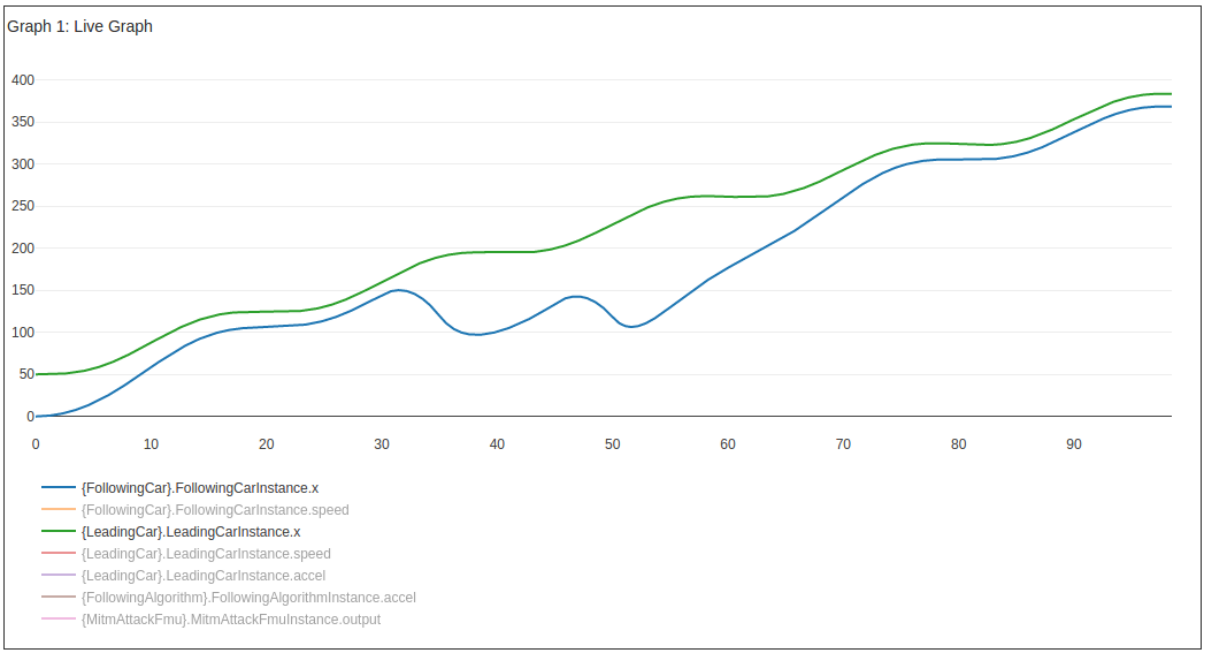


Figure 20: Grafico di posizione dei due veicoli nel caso di attacco multiplo in esame. Notare il non verificarsi di un incidente e il ritorno a convergenza.

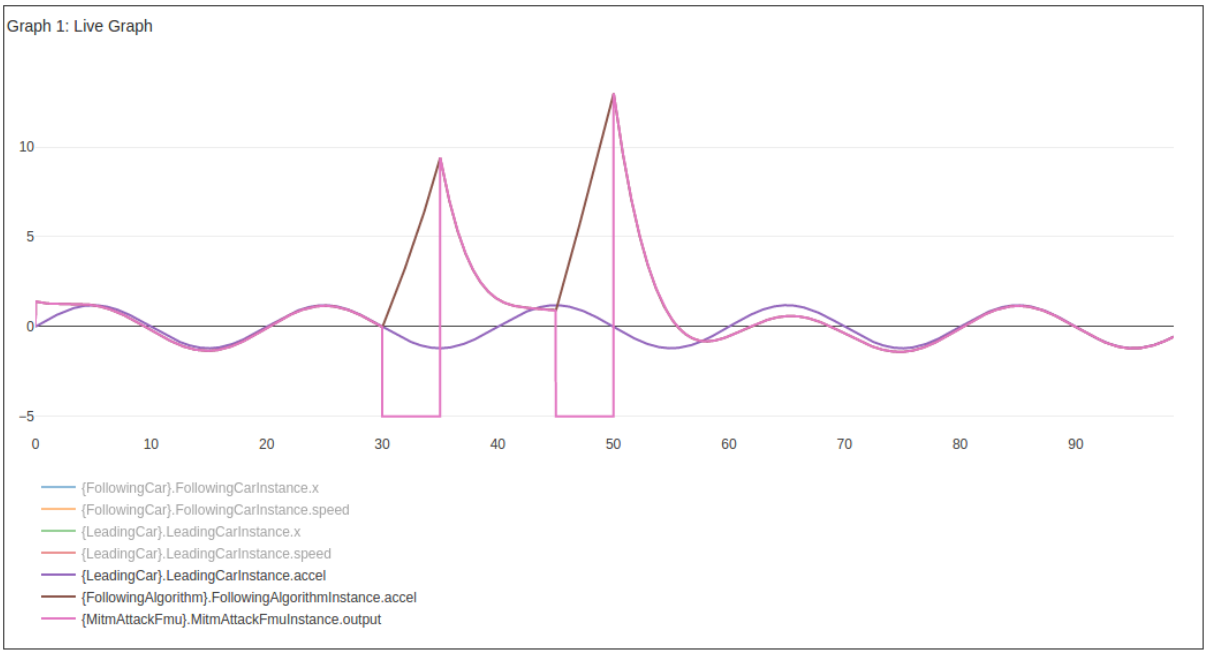


Figure 21: Grafico delle accelerazioni nel caso di attacco multiplo in esame.

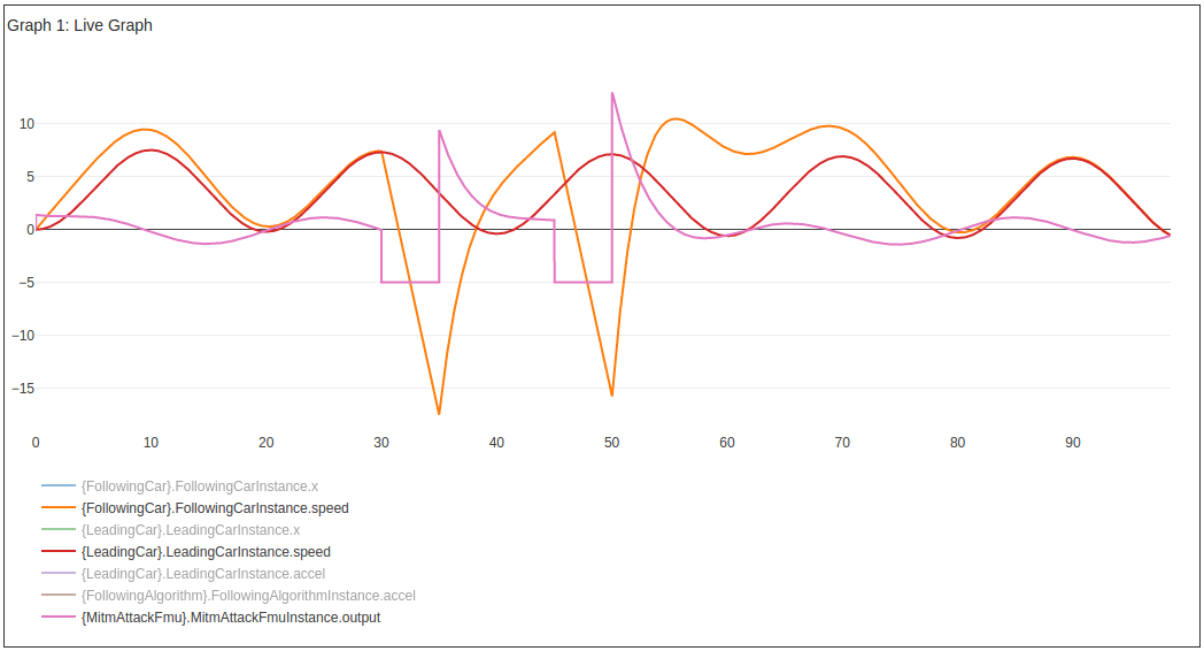


Figure 22: Grafico di velocità dei due veicoli nel caso di attacco multiplo in esame.

Osservando i grafici sopra descritti è possibile osservare come, nonostante il verificarsi di molteplici attacchi, la Following Car non crei alcun incidente. Inoltre è doveroso soffermare l'attenzione sulla tolleranza del sistema a questo tipo di attacco, al termine del quale la Following Car si avvicina nuovamente portandosi alla distanza di 15m dalla Leading Car.

Attacco con incidente L'obiettivo della presente co-simulazione è quello di andare ad individuare un attacco in cui la presenza di più occorrenze risulta essere un

elemento chiave nel verificarsi di un incidente fra i due veicoli. Di seguito sono riportate le configurazioni dell'attacco in esame.

- **Attack_occurrencies**: 2
- **Attack_duration**: 2s
- **Attack_time**: 30s
- **Attack_value**: $+2\ m/s^2$
- **Attack_distance**: 5s
- **Step_size**: 0.01s

Vengono ora riportati i risultati della co-simulazione nelle immagini seguenti.

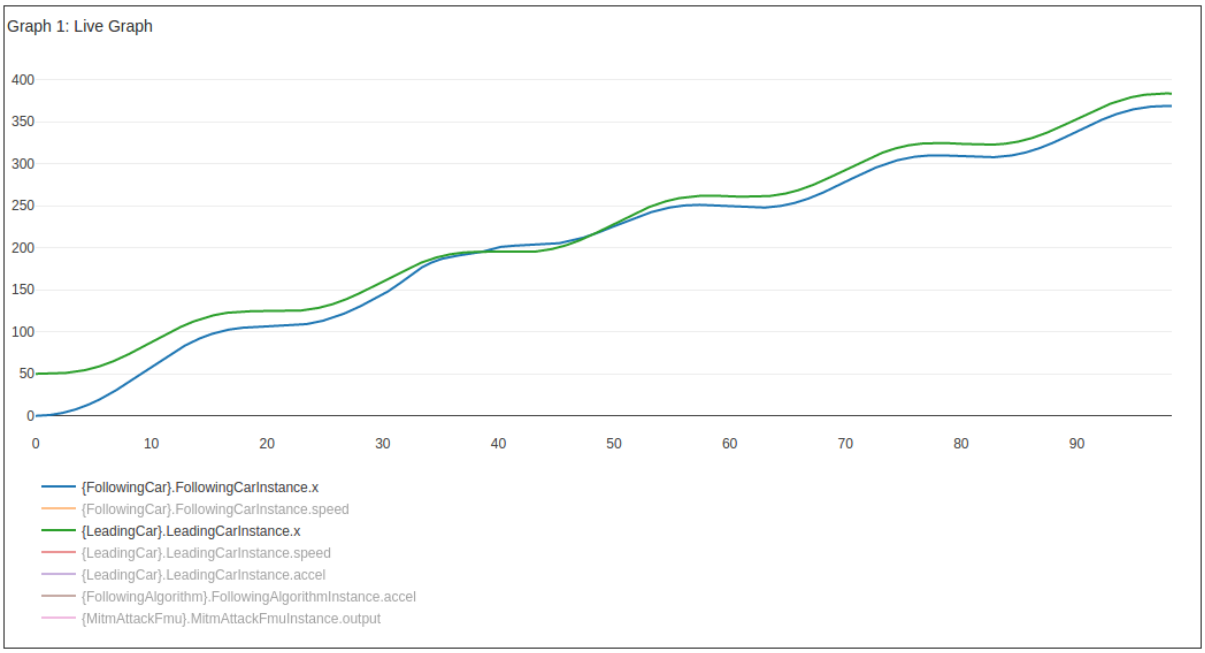


Figure 23: Grafico di posizione dei due veicoli nel caso di attacco multiplo in esame. Notare il verificarsi di un incidente.

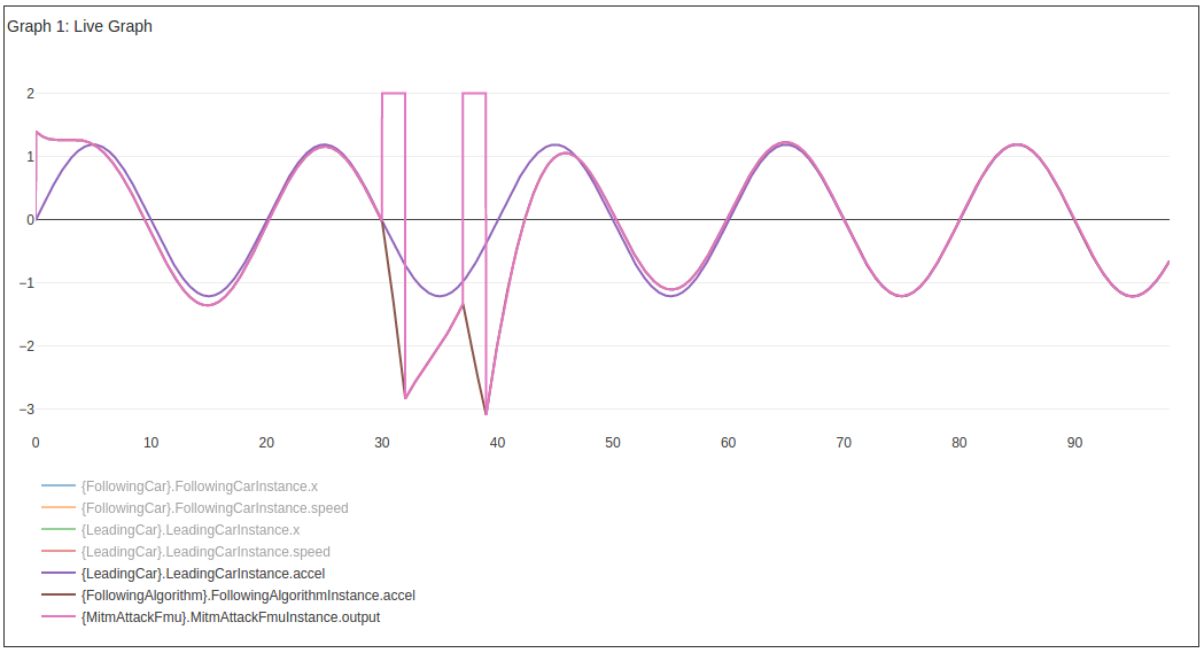


Figure 24: Grafico delle accelerazioni nel caso di attacco multiplo in esame.

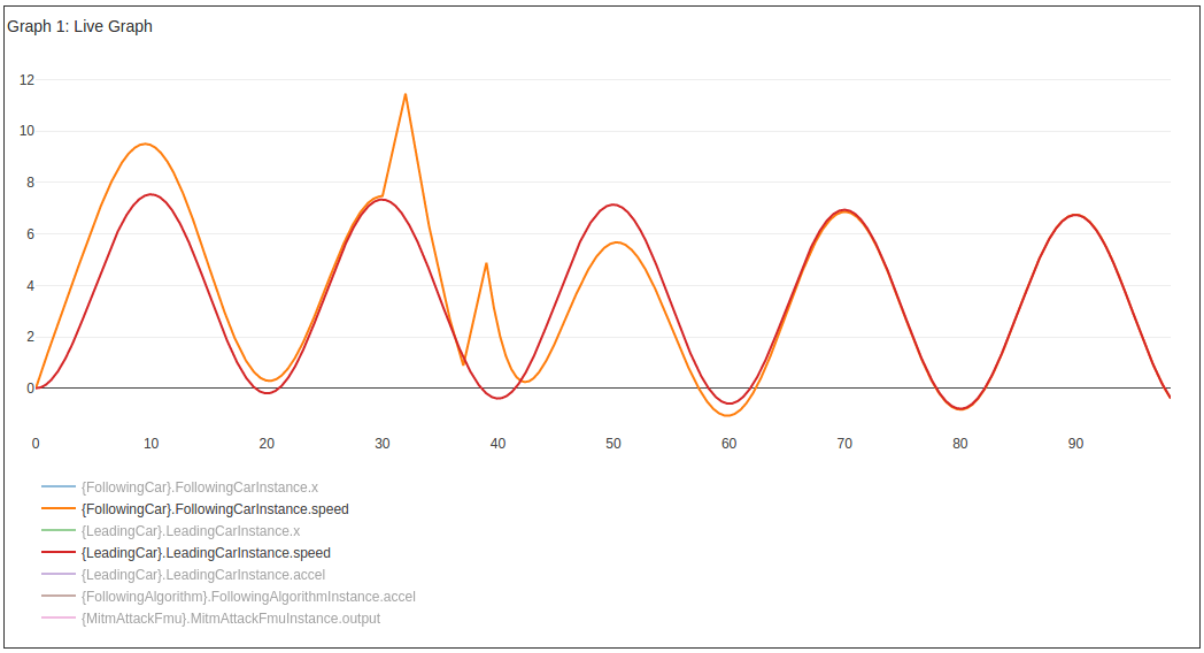


Figure 25: Grafico di velocità dei due veicoli nel caso di attacco multiplo in esame.

Osservando i grafici sopra descritti è possibile osservare come il secondo evento di attacco risulta fondamentale nel verificarsi dell'incidente. Senza questo secondo evento infatti la Following Car si sarebbe nuovamente distanziata dalla Leading Car così da raggiungere la distanza richiesta di 15m.

4.3 Attacco alla X

4.3.1 Attacco Semplice

Risultati Co-Simulazione Per cercare di dare un'interpretazione ai risultati del successivo studio verrà prima analizzato un caso d'esempio con i seguenti parametri:

- **attack_value**: 200m
- **attack_time**: 20s

Si ottiene il seguente grafico:

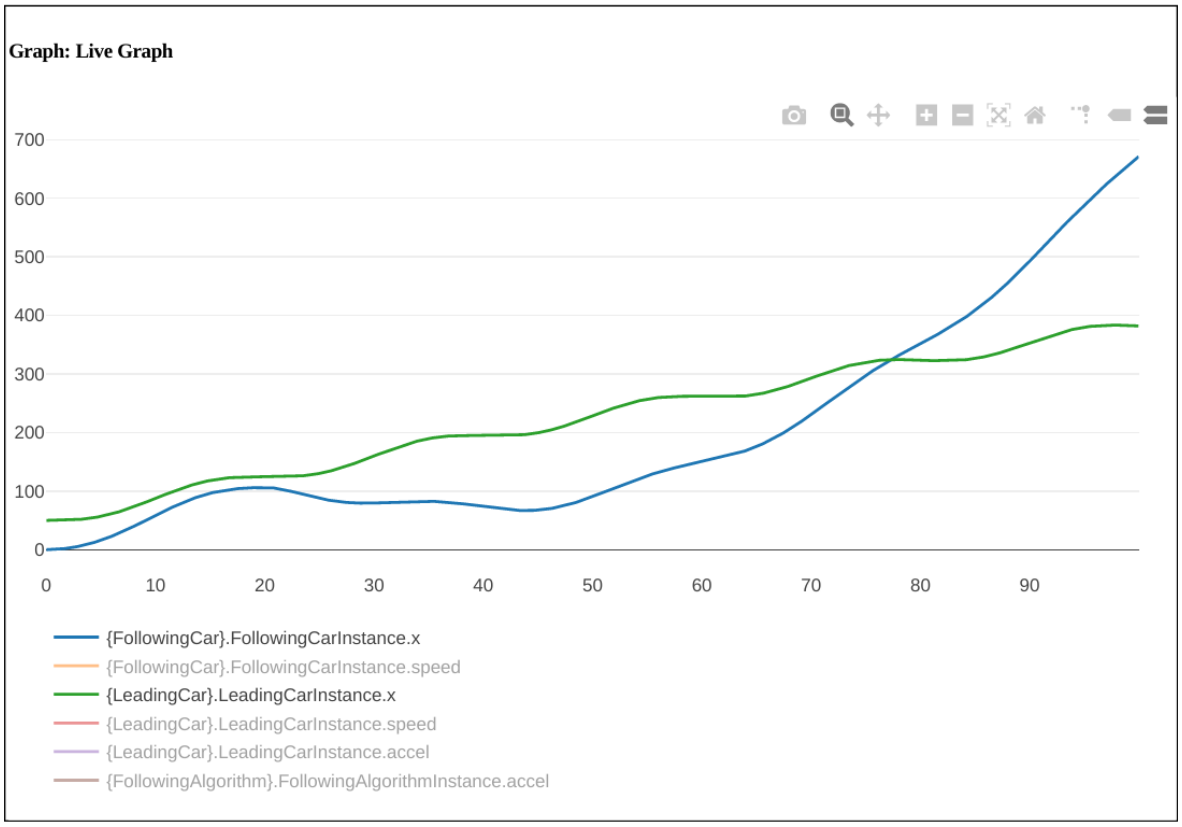


Figure 26: Posizione x della Leading Car (verde) e Following Car (blu)

Dal seguente risultato è possibile evincere tre differenti zone di comportamento della Following Car nel dominio dell'esperimento:

- **Primo Caso**: Da 0s a 20s, nel quale l'attacco non è ancora attivo e la following car tenderà ad avvicinarsi alla leading car mantenendo la distanza configurata di 15m.
- **Secondo Caso**: Da 20s a circa 40s, nel quale l'attacco avrà inizio, portando all'allontanamento delle due vetture. Questo è dovuto al fatto che la posizione della leading car è inferiore rispetto a quella dell'attack_value che rappresenta la posizione della following car vista dal Following Algorithm. Per questo motivo, il Following Algorithm calcolerà un'accelerazione negativa per la Following Car.

- **Terzo Caso**: Da 40s in poi, nel quale la Leading Car avrà superato il valore di attack_value e il Following Algorithm calcolerà un valore di accelerazione positiva per la Following Car. Questo caso porterà all'impatto tra le due vetture.

Nel dominio dell'esperimento la leading car tende ad andare "in avanti" continuamente con qualche oscillazione nella velocità, è facile intuire che **un incidente con questo tipo di attacco avrà luogo con alta probabilità per valori di attack value nel dominio dell'esperimento**, in quanto molto probabilmente la Leading Car supererà l'attack_value per un tempo significativo per permettere alla Following Car di superarla.

Risultati DSE E' stato studiato l'esito dell'attacco (INCIDENTE/NON INCIDENTE) andando a variare l'**attack_value** e l'**attack_time** con i seguenti parametri:

- **Attack_value**: [0m .. 200m] con step a 1
- **Simulation_time**: [50s, 100s]

I risultati ottenuti possono essere riassunti nella seguente tabella

Tempo di Simulazione [s]	Attack Value [m]	Risultato
50	[0, 149]	INCIDENTE
	[150, 199]	NO INCIDENTE
100	[0, 199]	INCIDENTE
	-	NO INCIDENTE

Come si può notare il tempo è una variabile importante per questo tipo di attacco, con un tempo sufficientemente alto dell'esperimento l'attacco ha sempre luogo nel dominio dell'esperimento.

4.4 Attacco Multiplo

Sono state individuate tre diverse configurazioni che danno luogo a quattro classi di risultati diversi:

- **Attack_occurrencies**: 3
- **Attack_duration**: 2s
- **Attack_time**: [10s, 30s, 50s, 70s]
- **Attack_value**: 200m
- **Attack_distance**: 5s
- **Step_size**: 0.01s

L'attacco pertanto avrà un pattern simile a livello temporale, la variabile è l'inizio dell'attacco stesso. I risultati degli esperimenti sono riassunti nella seguente tabella

Attack Time [s]	Distanza Minima [m]	Risultato
10	14.9326	NO INCIDENTE
30	14.9368	NO INCIDENTE
50	0.639284	NO INCIDENTE
70	-20.38	INCIDENTE

Le seguenti quattro figure vengono rappresentati i relativi grafici della posizione delle due auto.

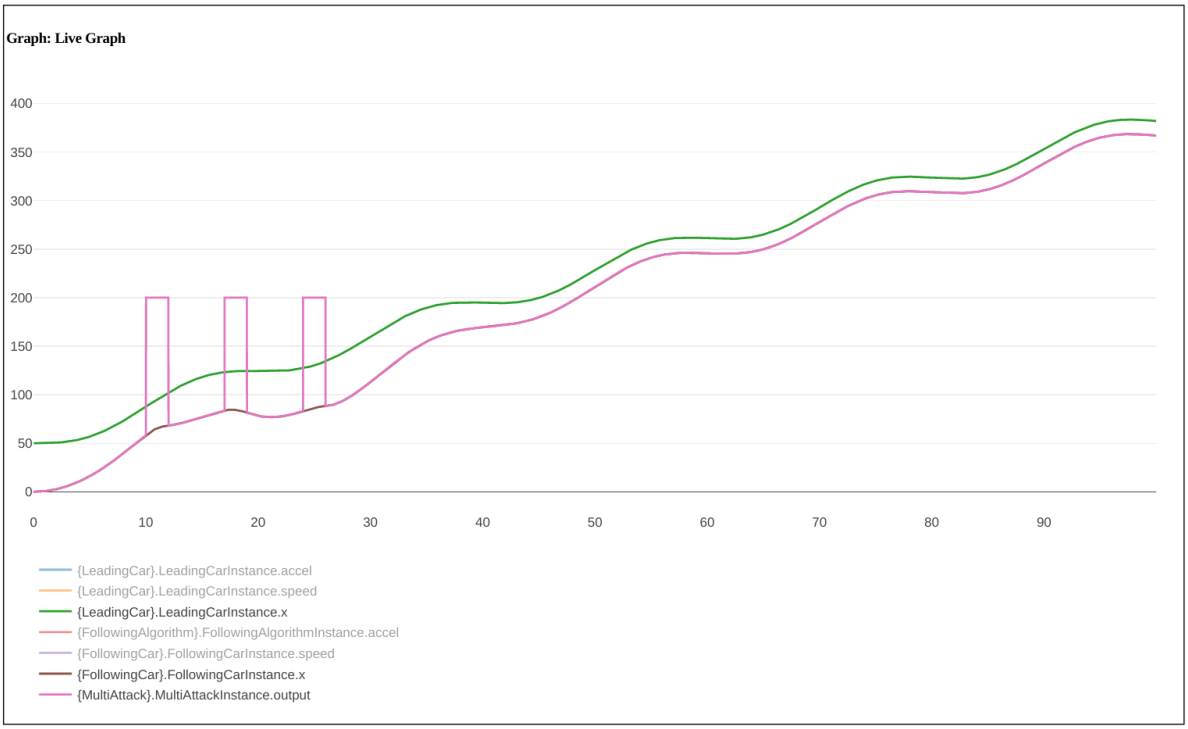


Figure 27: Grafico delle posizioni di Leading e Following Car e output dell'Attack_fmu per attack_time a 10.

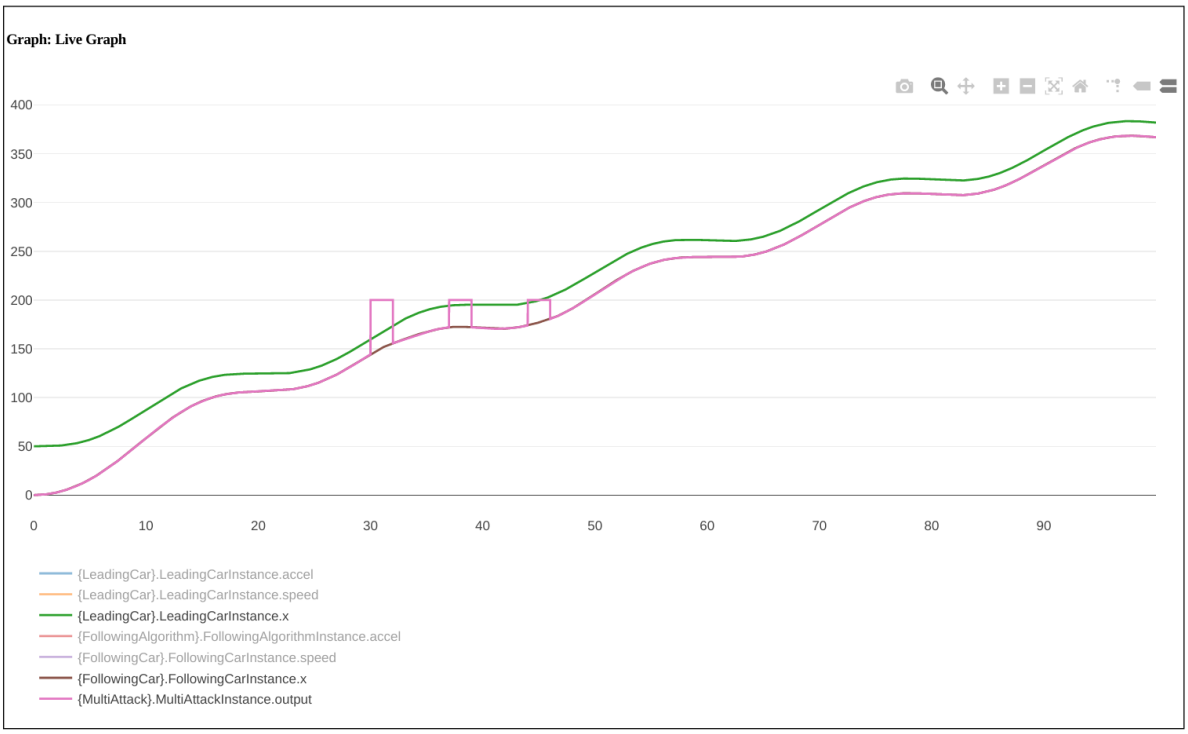


Figure 28: Grafico delle posizioni di Leading e Following Car e output dell'Attack_fmu per attack_time a 30.

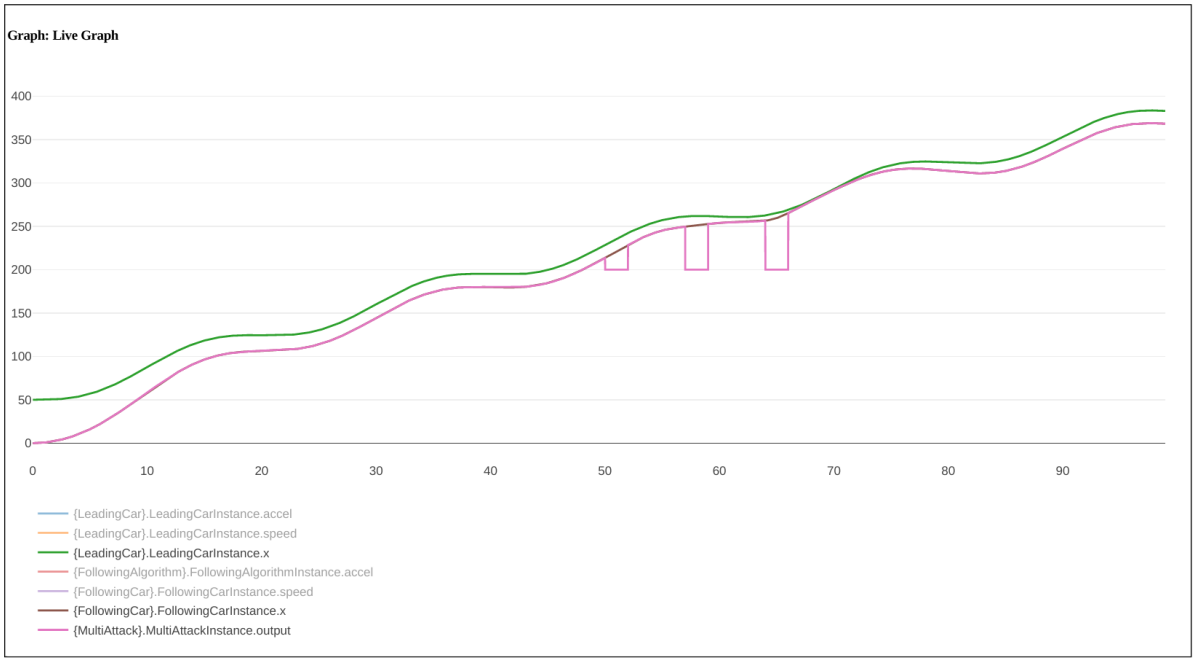


Figure 29: Grafico delle posizioni di Leading e Following Car e output dell'Attack_fmu per attack_time a 50.

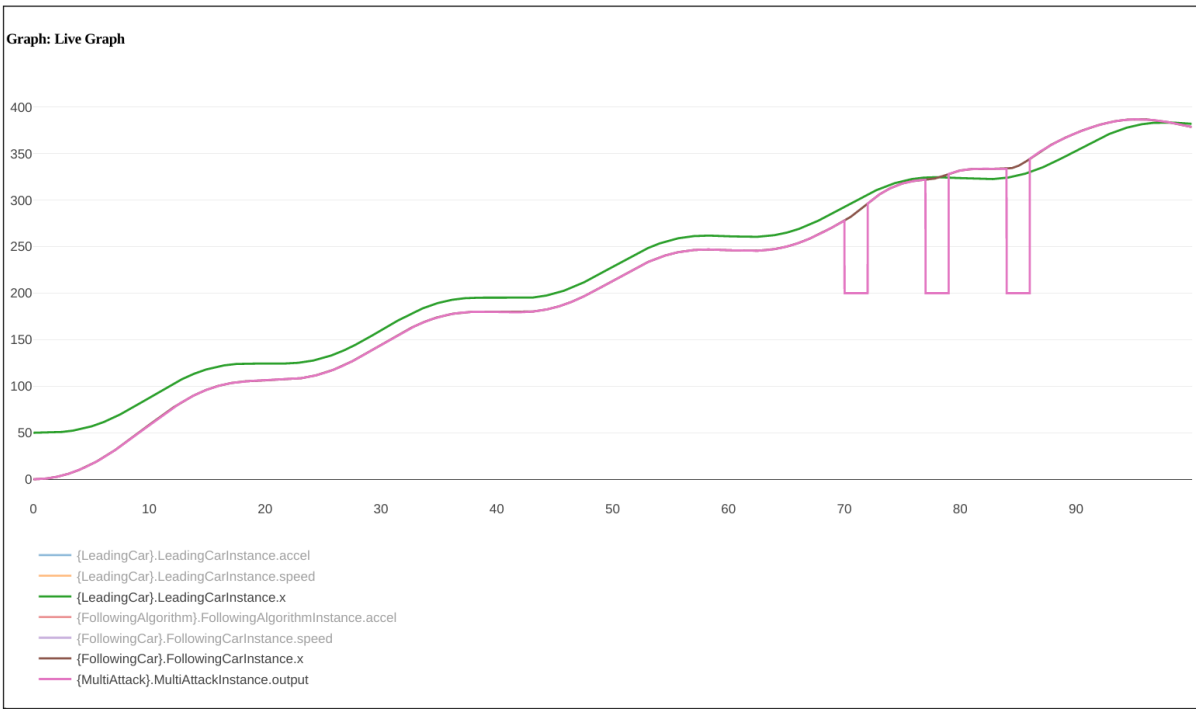


Figure 30: Grafico delle posizioni di Leading e Following Car e output dell'Attack_fmu per attack_time a 70.

Una semplice interpretazione di questi risultati si basa sul fatto che il Following Algorithm produce un'accelerazione maggiore nel caso la distanza tra le due auto sia maggiore. Premettendo che la distanza della Following Car vista dal Following Algorithm è fissa (per via dell'attacco in corso), con un tempo di inizio di attacco più elevato, la distanza tra la Leading Car e la Following Car assumerà valori più elevati. Questo porterà ad un'accelerazione in input alla Following Car maggiore negli intervalli di tempo in cui l'attacco avrà luogo. Per questo motivo con un attack time pari a 70s vi è un incidente.

5 — Conclusioni

A fronte dello studio riportato in questo documento risulta evidente come i casi di attacchi(tra Following Algorithm e Following Car) alla posizione e quelli all'accelerazione (con valore positivo) possano essere identificati come i casi più critici in quanto portano con estrema probabilità ad un incidente tra i veicoli.
Ad opinione degli autori di questo documento sarebbe opportuno investire risorse per contrastare queste casistiche rendendo il sistema più tollerante: ad esempio aggiungere ridondanza tra i collegamenti per individuare condizioni di attacco.
Attacchi all'accelerazione con valore pari a 0 m/s^2 risultano scaturire in comportamenti variabili a seconda del tempo di attacco.
Attacchi all'accelerazione con valori negativi risultano invece meno critici dal punto di vista degli incidenti, i quali risultano essere altamente improbabili nel dominio della simulazione.