

PROGETTO LABORATORIO SISTEMI OPERATIVI

ANNO ACCADEMICO 2023/24

Autori:

-Casula Carmelo, 7085019, carmelo.casula@edu.unifi.it

- Valdrighi Gabriele, 7048576, gabriele.valdrighi@edu.unifi.it

Data consegna: 03/02/2024

Istruzioni per la compilazione:

All'interno della cartella è presente un Makefile che compila tutti i file sorgente in eseguibili, per eseguirlo basta posizionarsi da terminale nella cartella e scrivere il comando "make".

Istruzioni per l'esecuzione:

Una volta compilato il progetto, si dovrà lanciare l'**hmi** scrivendo `"./hmi"` sul terminale insieme al tipo di esecuzione che si vorrà utilizzare (NORMALE o ARTIFICIALE).

Specifiche Hardware e Software:

Il progetto è stato realizzato tramite Visual Studio Code, la versione del compilatore gcc 11.4.0, il make 4.3 e la distribuzione Ubuntu Jammy 22.04.03

Il progetto è stato scritto, eseguito e testato su un MacBook Pro del 2019 con le seguenti specifiche:

Processore: 2,6 GHz Intel Core i7 6 core;

Schede Grafiche: AMD Radeon PRO 5300M 4GB, INTEL UHD Graphics 630 1536 MB;

Memoria principale: 16GB 2667 MHz DDR4;

Memoria secondaria: 512GB SSD;

È stato installato Ubuntu tramite VirtualBox versione 7.0.10r15879 a cui sono stati dati 4096 MB di memoria principale, 50GB di memoria secondaria, 4 processori, e 16MB di memoria video con scheda grafica VMSVGA.

Elementi facoltativi realizzati:

	Elemento Facoltativo	Realizzato (SI/NO)	Descrizione dell'implementazione con indicazione del metodo/i principale/i
1	Ad ogni accelerazione, c'è una probabilità di 10-5 che l'acceleratore fallisca. In tal caso, il componente throttle control invia un segnale alla Central ECU per evidenziare tale evento, e la Central ECU avvia la procedura di ARRESTO	SI	Il file throttleControl.c legge un numero dalla sorgente letta e lo modula a 100000, se è il numero 48925 manda il segnale SIGUSR1 a centralEcu.c che lo gestisce col signal handler erroreAccelerazione a riga 18
2	Componente "forward facing radar"	NO	
3	Quando si attiva l'interazione con park assist, la Central ECU sospende (o rimuove) tutti i sensori e attuatori, tranne park assist e surround view cameras	SI	Quando viene avviata la procedura di parcheggio a riga 201, viene fatta la kill() su tutti tranne BrakeByWire e SteerByWire per far azzerare la velocità per poi essere killati anche loro, tutto questo prima di avviare parkAssist
4	Il componente Park assist non è generato all'avvio del Sistema, ma creato dalla Central ECU al bisogno.	SI	Nella centralEcu.c a riga 224, viene generato solo quando viene avviata la procedura di parcheggio
5	Se il componente surround view cameras è implementato, park assist trasmette a Central ECU anche i byte ricevuti da surround view cameras.	SI	parkAssist con due write() separate (righe 32 e 35) manda prima i byte di SurroundViewCameras poi quelli di parkAssist, prima di fare la sleep(1)
6	Componente "surround view cameras"	SI	File "SurroundViewCameras.c"
7	Il comando di PARCHEGGIO potrebbe arrivare mentre i vari attuatori stanno eseguendo ulteriori comandi (accelerare o sterzare). I vari attuatori interrompono le loro azioni, per avviare le procedure di parcheggio.	SI	Ogni volta prima di accelerare/frenare (nella ecu righe 154 e 177) o prima/durante una sterzata (direttamente su SteerByWire righe 29 e 49), se viene letto "PARCHEGGIO" da

			inputForHMI, si interrompe l'azione in esecuzione (nella centralEcu righe 163 e 186, in steerByWire riga 44)
8	Se la Central ECU riceve il segnale di fallimento accelerazione da "throttle control", imposta la velocità a 0 e invia all'output della HMI un messaggio di totale terminazione dell'esecuzione	SI	Quando viene avviato il signal handler erroreAccelerazione, viene impostata la velocità a 0, viene scritto un messaggio di arresto a video e vengono terminati tutti i processi

Progettazione e implementazione:

hmi: prima di far partire tutti i processi, crea la pipe tra *inputForHMI* e *SteerByWire* per evitare errori di sincronizzazione tra questi, poi lancia prima la **centralEcu** tramite una *fork()*, la quale lancerà tutti gli altri processi, dopo, tramite una chiamata di sistema viene lanciato un secondo terminale sul quale viene eseguito **inputForHMI**, che permette di inserire i comandi in input (INIZIO, ARRESTO, PARCHEGGIO). Intanto si mette in attesa di terminazione del figlio (*centralEcu*) tramite la *wait()*. Dopo che la *centralEcu* termina, nel caso "PARCHEGGIO" venga letto dal file *frontCamera.data*, il secondo terminale rimane attivo, quindi cerca il pid di *inputForHMI* sul quale viene eseguita la *kill()* facendo terminare l'esecuzione.

InputForHMI: è il processo eseguito sul secondo terminale, apre subito *inputPipe* in sola scrittura con la *centralEcu* per mandargli ciò che viene scritto in input. Finché non viene scritto "INIZIO" viene richiesto di scriverlo per avviare la corsa, una volta scritto lo manderà tramite pipe alla *centralEcu*. Una volta partita la macchina, si collega alla pipe *steerHMI* in sola scrittura con *SteerByWire* per potergli comunicare l'arresto durante una sterzata. Finché non trova "PARCHEGGIO" chiede di scrivere "ARRESTO" o "PARCHEGGIO": nel caso di arresto lo invia alla *centralEcu* e se sta girando a destra o sinistra (lo si capisce aprendo il file *steer.log* e andando all'ultima posizione -2 per arrivare all'ultima lettera scritta, se è "A" vuol dire che sta sterzando, se no vuol dire che è in no action) lo manda pure a *SteerByWire*, nel caso di "PARCHEGGIO" esce dal ciclo e termina mandandolo alla *centralEcu* ed eventualmente a *SteerByWire*.

centralEcu: si occupa di creare le pipe con tutti i processi che va a generare tramite *fork()* e *exec()*, riceve tramite socket da *FrontWindshieldCamera* i dati letti dal file *frontCamera.data* e a seconda di essi, sblocca uno dei processi necessari tramite pipe. Se riceve un numero, controlla la velocità: se la velocità richiesta è maggiore o minore di quella attuale rispettivamente accelera e richiama *throttleControl* o rallenta e richiama *BrakeBywire*. Se durante queste operazioni riceve "ARRESTO" da *inputForHMI*, mette la

velocità a 0 e richiama *BrakeByWire*; se invece riceve “PARCHEGGIO”, interrompe l’operazione corrente che porterà ad avviare la procedura di parcheggio. Se riceve “DESTRA” O “SINISTRA” controlla se la velocità è 0, se sì prima di sterzare la incrementa di 5 richiama *throttleControl*, quindi richiama *SteerByWire* per sterzare. Se riceve “PERICOLO” o “ARRESTO” (quest’ultimo da *inputForHMI*) mette la velocità a 0 e lo segnala tramite pipe a *BrakeByWire*. Se riceve “PARCHEGGIO” (anche da *inputForHMI*) porta la velocità a 0 richiama *BrakeByWire*, termina tutti i processi, quindi avvia *parkAssist* e *SurroundViewCameras*: per 30 secondi prende 8 byte da *parkAssist* e li unisce a coppie di 2, se ognuno di essi corrisponde ai valori di errore, riavvia la procedura di parcheggio azzerando il contatore del ciclo, altrimenti termina uccidendo i processi attivi rimasti.

Di seguito la descrizione dei processi lanciati dalla **centralEcu**:

FrontWindshieldCamera: si collega alla socket con la **centralEcu**, una volta collegato, fino al termine della corsa una volta al secondo legge da *frontCamera.data* una riga alla volta. Per farlo legge un carattere alla volta finché non trova \n o EOF, quindi la manda alla *centralEcu* e viene scritta nel file *camera.log*;

throttleControl: apre la pipe *throttlePipe* in sola lettura con la *centralEcu*, controlla se *argv[1]* è “NORMALE” o “ARTIFICIALE” per la gestione dell’errore di accelerazione. Fino al termine della corsa, quando legge dalla *centralEcu*, legge un numero random dal file associato e lo modula per avere la probabilità di 10^{-5} . Se è uguale a un numero casuale scelto da noi compreso tra 0 e 99999 (48925), invia il segnale SIGUSR1 alla *centralEcu* tramite la *kill()*, altrimenti scrive nel file di log *throttle.log* la data attuale e “INCREMENTO 5”;

brakeByWire: apre la pipe *brakePipe* in sola lettura con la *centralEcu*, quando legge dalla *centralEcu*, controlla se la prima lettera è “F”, se lo è significa che deve frenare, quindi scrive nel file *brake.log* la data attuale e “DECREMENTO 5”, altrimenti significa che siamo in situazione di arresto, perciò scrive la data attuale e “ARRESTO AUTO”;

SteerByWire: questo processo apre la pipe *steerPipe* in sola lettura non bloccante con la *centralEcu* e apre allo stesso modo la pipe *steerHMI* con *inputForHMI*. Una volta al secondo, per evitare che non avvenga un arresto precedente durante una sterzata successiva, legge da *inputForHMI*; controlla che non abbia ricevuto ordine di sterzata tramite pipe: se non lo riceve scrive nel file *steer.log* “NO ACTION”. Altrimenti, per 4 secondi, una volta al secondo prova a leggere da *inputForHMI*: se legge e non è “PARCHEGGIO” scrive “NO ACTION” e si blocca per un secondo per far sì che venga eseguito l’arresto dell’auto, quindi riscrive “NO ACTION” e si blocca per un altro secondo per far sì che l’auto inizi ad accelerare prima di riprendere la sterzata, altrimenti scrive nel

file *steer.log* "STO GIRANDO A " + la direzione ricevuta e se per caso ha letto "PARCHEGGIO" esce dal ciclo;

parkAssist: viene generato solo quando la *centralEcu* riceve la stringa “PARCHEGGIO” da *FrontWindshieldCamera* o da *inputForHMI*. Apre *parkPipe* con la *centralEcu* in sola scrittura e *surroundPipe* in sola lettura con *SurroundViewCameras*, una volta al secondo invia alla *centralEcu* gli 8 byte letti da **SurroundViewCameras**, poi legge da */dev/urandom* o da *urandomArtificiale.binary* (a seconda della tipologia di esecuzione che si è scelta) 8 byte esadecimali in un buffer e li invia alla *centralEcu*. Infine, scrive su *assist.log* gli 8 byte letti dalla sorgente scelta.

SurroundViewCameras: apre *surroundPipe* in sola scrittura con *parkAssist*, una volta al secondo finché la macchina non è parcheggiata, legge 8 byte random dalla sorgente scelta, i quali vengono mandati a *parkAssist* e scritti sul file *cameras.log*.

Esecuzione:

Di seguito un avvio di esecuzione in modalità normale:

The screenshot displays the macOS desktop environment during a car simulation project. The top status bar indicates the date and time as 'gio 1 feb 18:20'. The dock on the left side contains icons for various applications, including Firefox, Telegram, Finder, Visual Studio Code, and the Terminal. The main workspace is divided into three panes:

- Explorer (Left):** Shows the project structure with files like 'centralEcu.c', 'throttle.log', and 'inputForHMI.c'.
- Central Pane:** Displays the contents of 'throttle.log', which contains a table of speed increments over time.
- Right Pane:** Shows the terminal output of the './hmi NORMALE' command. The terminal text includes instructions to start the race by typing 'INIZIO' and to stop by typing 'ARRESTO' or 'PARCHEGGIO'.

The terminal output shows the following sequence of events:

```
drigo@drigoMac: ~/Scrivania/progetto
drigo@drigoMac:~/Scrivania/progetto$ ./hmi NORMALE
Per avviare la corsa digita INIZIO sul terminale per l'input
Avvio la corsa
Arrivo a 50 km/h, accelero di 50 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 5 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 10 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 15 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 20 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 25 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 30 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 35 km/h
```

Inserimento dell'Input "ARRESTO" durante una sterzata, notare sul terminale che viene fermata la sterzata, riavviata la corsa per poi riprendere la sterzata:

The screenshot shows the Visual Studio Code interface with the 'steer.log' file open. The log file contains the following entries:

```

64 NO ACTION
65 NO ACTION
66 NO ACTION
67 STO GIRANDO A DESTRA
68 STO GIRANDO A DESTRA
69 STO GIRANDO A DESTRA
70 STO GIRANDO A DESTRA
71 NO ACTION
72 NO ACTION
73 NO ACTION
74 NO ACTION
75 NO ACTION
76 NO ACTION
77 NO ACTION
78 NO ACTION
79 NO ACTION
80 NO ACTION
81 NO ACTION
82 STO GIRANDO A DESTRA
83 STO GIRANDO A DESTRA
84 NO ACTION
85 NO ACTION
86 STO GIRANDO A DESTRA
87 STO GIRANDO A DESTRA
88 NO ACTION
89 NO ACTION
90 NO ACTION
91

```

The terminal window shows the following output:

```

Arrivo a 50 km/h, accelero di 10 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 45 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 50 km/h
Mantengo la velocità di 50 km/h
Arrivo a 60 km/h, accelero di 10 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 60 km/h
Mantengo la velocità di 60 km/h
Sto girando a 60 km/h a DESTRA
Arrivo a 50 km/h, freno di 10 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 50 km/h
Mantengo la velocità di 50 km/h
Sto girando a 50 km/h a DESTRA
ARRESTO AUTO, VELOCITÀ = 0 km/h
Riprendo la corsa...
Arrivo a 50 km/h, accelero di 50 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 5 km/h
Riprendo la sterzata a DESTRA
Incremento la velocità di 5 km/h, VELOCITÀ = 10 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 15 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 20 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 25 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 30 km/h

```

The terminal window also shows the following prompt:

```

Per avviare la corsa digita INIZIO
INIZIO
Digitare ARRESTO oppure PARCHEGGIO per continuare
ARRESTO
Digitare ARRESTO oppure PARCHEGGIO per continuare

```

Inserimento dell'Input "PARCHEGGIO" sempre durante una sterzata, si nota come non termini la sterzata nel file di log:

The screenshot shows the Visual Studio Code interface with the 'steer.log' file open. The log file contains the following entries:

```

166 NO ACTION
167 NO ACTION
168 NO ACTION
169 NO ACTION
170 NO ACTION
171 NO ACTION
172 NO ACTION
173 NO ACTION
174 NO ACTION
175 STO GIRANDO A DESTRA
176 STO GIRANDO A DESTRA
177 NO ACTION
178 NO ACTION
179 NO ACTION
180 NO ACTION
181 NO ACTION
182 NO ACTION
183 NO ACTION
184 NO ACTION
185 NO ACTION
186 NO ACTION
187 NO ACTION
188 NO ACTION
189 NO ACTION
190 NO ACTION
191 NO ACTION
192 NO ACTION
193 NO ACTION
194 NO ACTION
195 NO ACTION
196

```

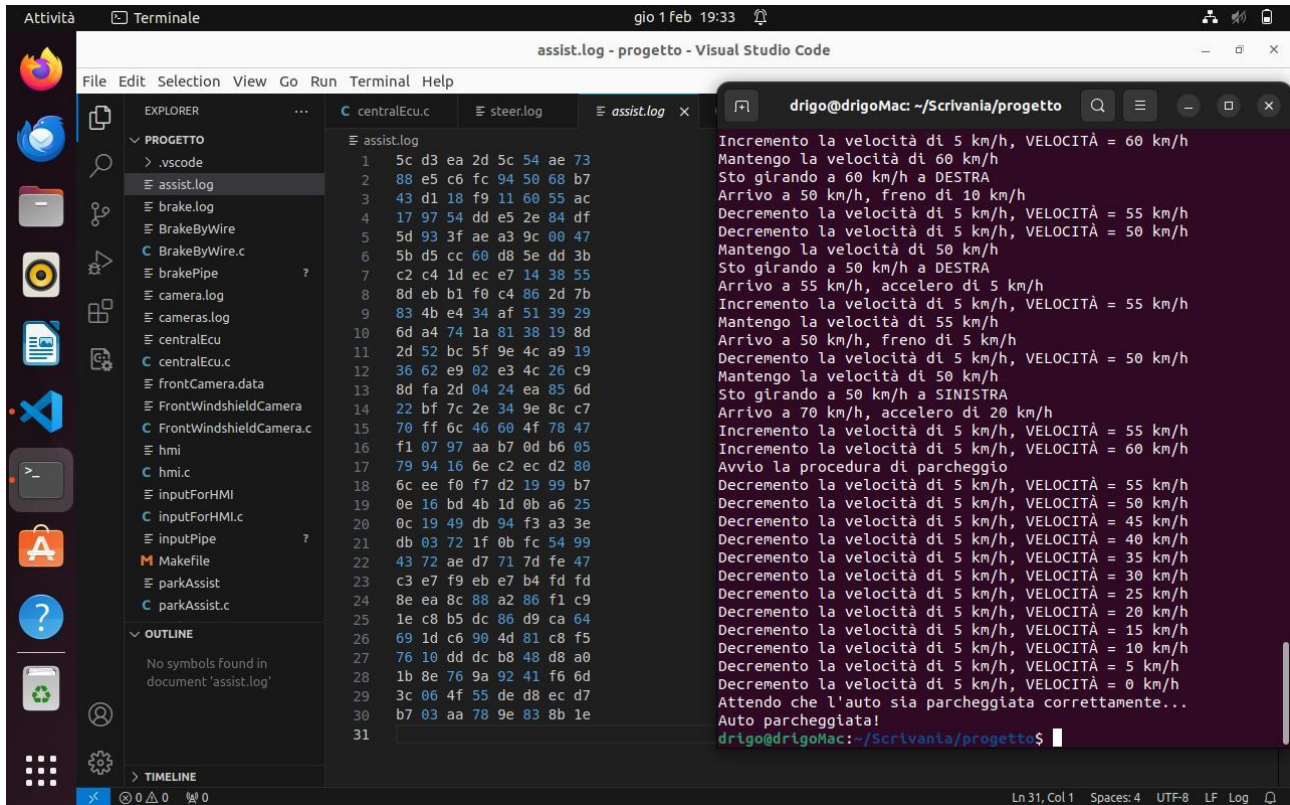
The terminal window shows the following output:

```

Mantengo la velocità di 50 km/h
Arrivo a 90 km/h, accelero di 40 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 60 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 65 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 70 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 75 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 80 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 85 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 90 km/h
Mantengo la velocità di 90 km/h
Sto girando a 90 km/h a DESTRA
Avvio la procedura di parcheggio
Decremento la velocità di 5 km/h, VELOCITÀ = 85 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 80 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 75 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 70 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 65 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 60 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 50 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 45 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 40 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 35 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 30 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 25 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 20 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 15 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 10 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 5 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 0 km/h
Attendo che l'auto stia parcheggiata correttamente...
Auto parcheggiata!
drigo@drigoMac:~/Scrivania/progetto$

```


Inserimento dell'Input "PARCHEGGIO" durante un'accelerazione, anche qua non finisce l'azione in corso e inizia a parcheggiare concludendo la corsa (di fianco i byte letti da *parkAssist*):



```
1 5c d3 ea 2d 5c 54 ae 73
2 88 e5 c6 fc 94 50 68 b7
3 43 d1 18 f9 11 60 55 ac
4 17 97 54 dd e5 2e 84 df
5 5d 93 3f ae a3 9c 00 47
6 5b d5 cc 60 d8 5e dd 3b
7 c2 c4 1d ec e7 14 38 55
8 8d eb b1 f0 c4 86 2d 7b
9 83 4b e4 34 af 51 39 29
10 6d a4 74 1a 81 38 19 8d
11 2d 52 bc 5f 9e 4c a9 19
12 36 62 e9 02 e3 4c 26 c9
13 8d fa 2d 04 24 ea 85 6d
14 22 bf 7c 2e 34 9e 8c c7
15 70 ff 6c 46 60 4f 78 47
16 f1 07 97 aa b7 0d b6 05
17 79 94 16 6e c2 ec d2 80
18 6c ee f0 f7 d2 19 99 b7
19 0e 16 bd 4b 1d 0b a6 25
20 0c 19 49 db 94 f3 a3 3e
21 db 03 72 1f 0b fc 54 99
22 43 72 ae d7 71 7d fe 47
23 c3 e7 f9 eb e7 b4 fd fd
24 8e ea 8c 88 a2 86 f1 c9
25 1e c8 b5 dc 86 d9 ca 64
26 69 1d c6 90 4d 81 c8 f5
27 76 10 dd dc b8 48 d8 a0
28 1b 8e 76 9a 92 41 f6 6d
29 3c 06 4f 55 de d8 ec d7
30 b7 03 aa 78 9e 83 8b 1e
31
```

```
Incremento la velocità di 5 km/h, VELOCITÀ = 60 km/h
Mantengo la velocità di 60 km/h
Sto girando a 60 km/h a DESTRA
Arrivo a 50 km/h, freno di 10 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 50 km/h
Mantengo la velocità di 50 km/h
Sto girando a 50 km/h a DESTRA
Arrivo a 55 km/h, accelero di 5 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Mantengo la velocità di 55 km/h
Arrivo a 50 km/h, freno di 5 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 50 km/h
Mantengo la velocità di 50 km/h
Sto girando a 50 km/h a SINISTRA
Arrivo a 70 km/h, accelero di 20 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Incremento la velocità di 5 km/h, VELOCITÀ = 60 km/h
Avvio la procedura di parcheggio
Decremento la velocità di 5 km/h, VELOCITÀ = 55 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 50 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 45 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 40 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 35 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 30 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 25 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 20 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 15 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 10 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 5 km/h
Decremento la velocità di 5 km/h, VELOCITÀ = 0 km/h
Attendo che l'auto sia parcheggiata correttamente...
Auto parcheggiata!
drigo@drigoMac: ~/Scrivania/progetto$
```

L'esecuzione in modalità NORMALE o ARTIFICIALE è fatta in modo uguale, cambia solo la sorgente di lettura per *parkAssist*, *throttleControl* e *SurroundViewCameras*.