MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
CATEDRA CALCULATOARE

Validarea manevrei de virare la stânga

-Documetație-

Autor: Pop Ruxandra Maria
Grupa: 30643

Profesor îndrumător: Radu Răzvan Slăvescu

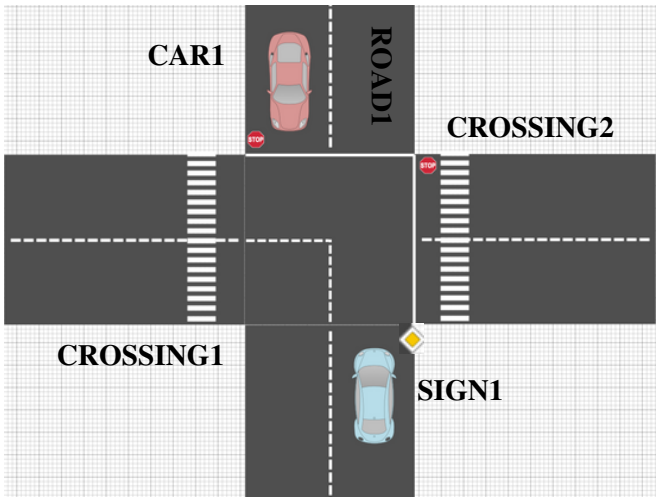
Cuprins

1. Domeniul aplicației și limitele sale

Conducerea autonomă are un domeniu de aplicabilitate foarte larg. Principiul de bază fiind conducerea în siguranță.
Aplicația de față conține un fisier numit Driver Agent care reprezintă creierul unui automobil autonom luând ca date de intrare înformații de la diferiți senzori, camere, sistem de navigație și acționând mai departe ca un controller care transmite decizia luată spre vehicul.
Structura aplicației constă din 3 module: MAIN, PERCEPT-MANAGER, DRIVER-AGENT. Modulul MAIN mută focusul dintre celeltate două module. Modulul PERCEPT-MANAGER actualizează timpul curent, sterge percepțiile anterioare și le citeste pe cele curente din fisierul dat ca parametru. Modulul DRIVER-AGENT este cel care implementează și validează manevrele, în cazul nostru virajul la stânga. Cele trei module se execută într-o buclă teoretic infinită, de aceea trebuie să specificam numarul de reguli pe care dorim să le executăm (?*totalNrRules*).
Manevrele cu privire la care agentul trebuie să ia decizia de viraj la stânga este specificată în fiș ierul maneuverValidityASK.clp.

2. Scenarii tratate

2.1 Intersecție fără semafor și cu două treceri de pietoni



EV1

Descriere scenariu

Acest scenariu prezintă o intersecție cu 2 drumuri fară prioritate și unul cu prioritate, nesemaforizată. Un drum fară prioritate se află în fața celui principal, iar celălalt în partea dreaptă a drumului principal, ambele având un indicator de oprire în intersecție. Conducătorul auto care trebuie să ia decizia de viraj la stânga, se află pe drumul principal (masina albastră), pe drumul fară prioritate se afla o mașină roz, care in diferite momente de timp are întenții diferite de a-și continua traseul (viraj la stânga, dreapta, mers în față). Consideram ca la fiecare moment de timp masina roz este de fapt o altă masină cu o intenție total diferită de cea precedentă. În intersecție mai se găsesc treceri de pietoni, una pe drumul principal (în stânga mașini autonome) și una pe un drum neprioritar (în partea dreaptă a mașinii autonome). În anumite momente de timp pe aceste treceri se găsesc pietoni aflați în traversare.

Percepții

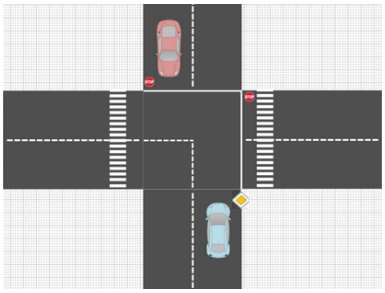
OBEJCT	NAME	VALUE
ev1	isa	eveniment
crossing1/2	isa	crossing
crossing1/2	partof	ev1
crossing1/2	direction	left/right
crossing1/2	state	free/busy
sign1	isa	sign
sign1	partof	ev1
sign1	type	priority
sign1	direction	ahead
road1	isa	road
road1	partof	ev1
road1	direction	ahead
car1	isa	car
car1	partof	ev1
car1	on	road1
car1	intention	ahead/right/left

Descriere percepții

Valorile folosite pentru percepțiile de tip isa sunt eveniment, crossing,sign, road, car, aceastea indica tipul obiectelor care pot fi observate de masina autonomă. Percepțiile de tip partof au fost folosite pentru a indica apartenența obiectelor observate la un eveniment. Celelalte percepții sunt:

- direction - poate avea valorile left, right, ahead și este folosită pentru a indica direcția pe care se află un anumit obiect, relativ la masina autonomă.
- state - poate avea valorile free/busy și este folosită pentru a indica faptul ca pe trecerea de pietoni se află sau nu pietoni angajații în trecere.
- type - indică tipul indicatorului.
- on - indică faptul ca o masină se află pe un anumit drum.
- intention - poate avea valorile aheah/right/left și indica faptul ca o masină semnalizează intenția unei anumite direcții de mers.

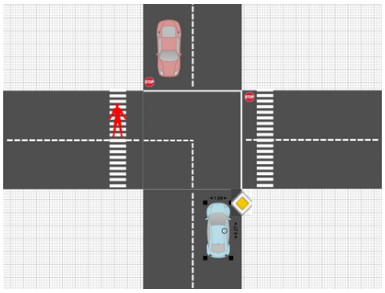
T1



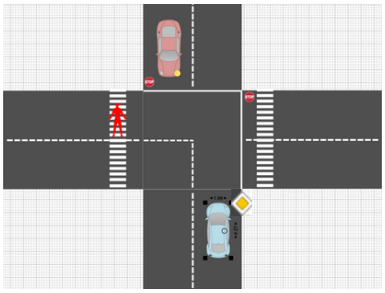
Metrica

Agentul auto va lua decizia de a vira la stânga indiferent de intenția masinii roz, pentru ca trecerea de pietoni din stânga acestuia este liberă, iar daca masina roz si-ar continua deplasare, i-ar incurca virajul masini autonome care se află pe drumul cu prioritate.

T2



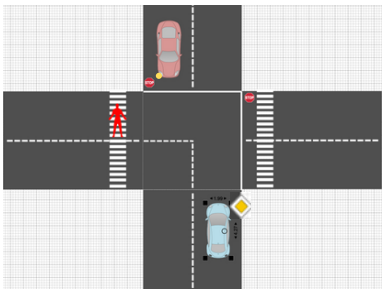
T5



Metrica

Agentul auto va lua decizia de a opri înaintea intersecției, desi se află pe drumul cu prioritate, pentru a nu bloca intersecția. Această decizie este luată ținând cont de faptul că pe direcția sa de deplasare se află o trecere de pietoni care în momentul de față nu este liberă și de faptul că pe drumul fără prioritate se află o masină a cărei întenție este de a merge în față (T2)/ stânga (T5) , drumul ei fiind neblocat. Momentul de timp T5 este condiționat și de faptul că pe trecerea din dreapta sa nu se află pietoni.

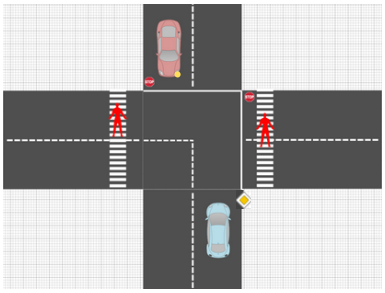
T3



Metrica

Pe drumul fără prioritate se afla o masină a cărei intenție este de a vira la dreapta, în acest caz agentul auto va vira la stânga deoarece dacă masină roz ar vira, s-ar afla pe drum în față masini autonome deoarece pe trecerea din stânga se află pietoni. Astfel dacă masina roz se află în fața celei autonome i-ar încurca deplasarea acesteia, și inseamnă că nu a respsctat indicatorul de stop.

T4



Metrica

Agentul auto va vira la stânga, deoarece trecerea din dreapta sa este ocupată iar masina de pe drumul fără prioritatate are intenția de a face stânga, blocând astfel întersecția.

Output

```
PERCEPT-MANAGER: timp = 1
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 2
      AGENT left-turn-maneuver prohibited
PERCEPT-MANAGER: timp = 3
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 4
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 5
      AGENT left-turn-maneuver prohibited
```

Faptul că agentul poate vira la stânga este marcat prin valoarea "allowed" a manevrei left-turn-maneuver.
Iar in caz contrar manevra este marcată prin valoarea "prohibited".
Fiecare decizie luată a fost explicată mai sus.

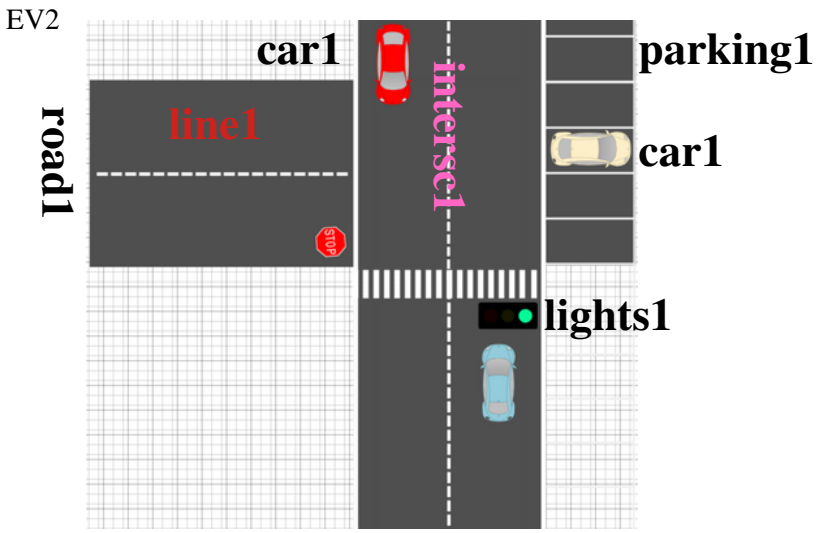
Raport de performanță

S-a rulat aplicatia de 5 ori și s-a înregistrat timpul de start si timpul de sfarșit, iar rezultatele au fost:

TIMP	MEDIA	SIGMA
T1	0,000044269561768	0,000009826779734930498
T2	0,000060272216797	0,000008455292248957564
T3	0,000041742335327	0,000023478772826834251
T4	0,000060987472494	0,000010566543584687269
T5	0,000048017501831	0,000023329610658349035

- Media aritmetică a timpilor de execuție si deviatia standard este calculata in secunde (sec)

2.2 Intersecție cu semafor, drum cu parcare laterală



Descriere scenariu

Acest scenariu prezintă un drum cu prioritate semaforizat pe care se află agentul auto, pe marginea acestui drum se află zone de parcare. Pe unul dintre aceste locuri de parcare se află un autovehicul care are sau nu întenția de a iesi din parcare. Totodată la un anumit moment de timp pe partea opusă de drum poate să apară o masină a cărei intenție este de a vira la stânga, pentru a intra în ultima parcare. În funcție de cât de aproape se află masina rosie de linie 1, agentul auto va decide dacă poate sau nu vira la stânga.

Percepții

OBEJCT	NAME	VALUE
ev2	isa	eveniment
lights1	isa	lights
lights1	partof	ev2
lights1	direction	ahead
lights1	color	green/red
lights1	proximity	0
parking1	isa	parking
parking1	partof	ev2
parking1	direction	right
parking1	proximity	1
car1	isa	car
car1	partof	ev2
car1	on	parking1/intersection1
car1	intention	back/none/left
car1	height	4/3
car1	distention_from_lights1	1/2
road1	isa	road
road1	partof	ev2

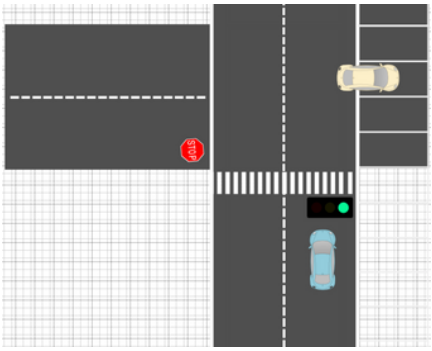
road1	direction	left
lane1	isa	line
lane1	partof	ev2
lane1	width	3
intersection1	isa	intersection
intersection1	partof	ev2
intersection1	direction	ahead
intersection1	height	6

Descriere percepții

Valorile folosite pentru percepțiile de tip isa sunt eveniment, car, lights, intersection, lane, road parking, aceastea indica tipul obiectelor care pot fi observate de masina autonomă. Percepțiile de tip partof au fost folosite pentru a indica apartenența obiectelor observate la un eveniment. Celelalte percepții sunt:

- direction - poate avea valorile left, right, ahead și este folosită pentru a indica direcția pe care se află un anumit obiect, relativ la masina autonomă.
- proximity -poate avea valorile 0,1 și arată cat de aproape se află obiectul față de agentul auto, aveasta percepție este necesare pentru a indica faptul că parcarea pe care se află masina se află după semafor.
- color - poate avea culorile red și green, fiind folosit pentru a indica culoarea semaforului.
- on - indică faptul ca o masină se află în intersecție sau în parcare.
- intention - poate avea valorile back/none/left și indica faptul ca o masină semnalizează intenția virajului la stânga sau intenția de a da cu spatele sau nu, pentru a iesi din parcare.
- height - reprezintă lungimea masini sau intersecției, și este necesară pentru a vedea dacă agentul auto are loc sa vireze la stânga pe lane1.
- distantion_from_lights1 - reprintă distanța dintre masina care vrea să intre in parcare și semafor, este folosită tot pentru a determina dacă agentul auto are spațiu pentru a intra pe lane1.Valoarea acesteia poate fi egală sau mai mare decât 1.
- width - reprezintă lațimea lui lane1, folosită pentru acelasi scop ca percepțile de mai sus.

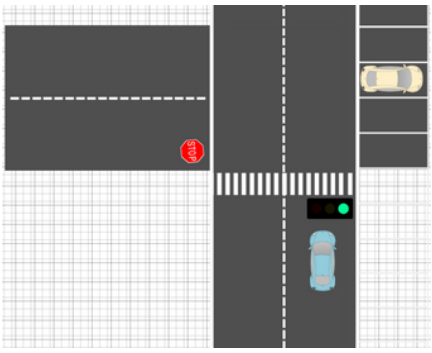
T1



Metrica

Agentul auto va lua decizia de a se opri pentru a-i face loc conducatorului auto să părăsească parcarea. Această decizie este luată de faptul că conducatorul mașinii galbene deja a depasit linia, aflându-se în intersecție și blocând banda pe care agentul auto vrea să meargă la stânga.

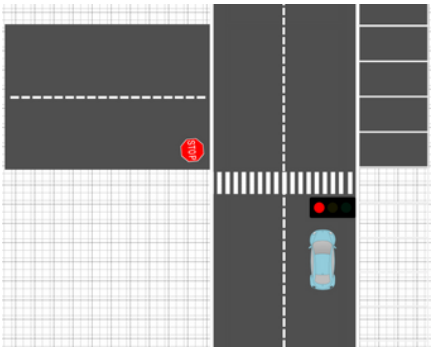
T2



Metrica

Agentul auto va face stânga deoarece semaforul este verde, iar intersecția nu este blocată (conducătorul mașini galbene nu prezintă întenția de a ieși din parcare).

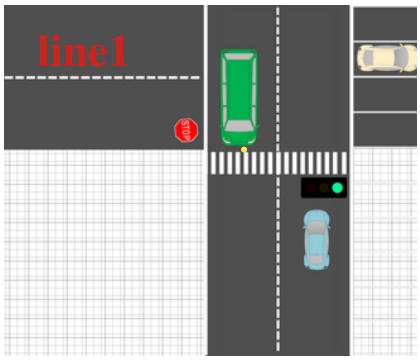
T6

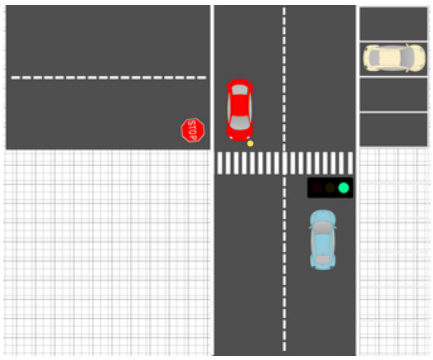


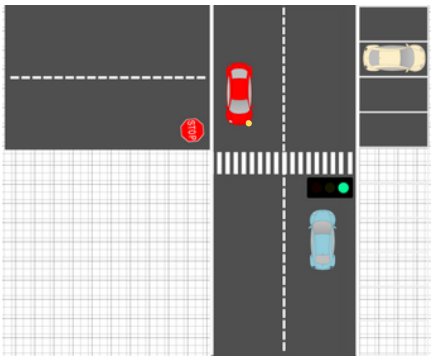
Metrica

Agentul auto nu va realiza virajul la stânga pentru ca semaforul are culoarea roșie.

T3 T4 T5







Metrica

În cazul acestor 3 momente de timp, fiecare masină vrea să facă stânga pentru a intra în ultima parcare.
Agentul auto care se află în masina albastra va decide dacă va face stânga pe line1, în funcție de spațiu ramăs în intersecție pentru a se încadra pe banda respectivă. Pentru validarea manevrei trebuie să se respecte 2 reguli:

1. diferența dintre lungimea intersecției și lungimea mașini care vrea să intre în parcare trebuie să fie mai mică sau egală cu latimea bandei 1, pentru ca nici o parte din line1 să nu fie ocupată de către autovehicul, astfel agentul auto să aibe suficient spațiu pentru a vira.
2. distanța dintre masina din intersecție și semafor să fie egala cu 1, pentru a determina că masină care vrea să intre in parcare se încadrează numai pe banda2, neocupând spațiu din line1.

Analizând aceste reguli putem observa ca pentru T3 regula 1 nu este satisfacută astfel agentul auto nu poate vira la stânga și trebuie să astepte ca masina verde să între în parcare pentru a-și continua drumul.
Pentru T4 ambele reguli sunt îndeplinite, astfel agentul auto ia decizia de a vira la stânga.
În cazul lui T5 regula 2 nu este satisfăcută, masină roșie aflându-se la o distanță mai mare decât 1.

Output

```
PERCEPT-MANAGER: timp = 1
      AGENT left-turn-maneuver prohibited
PERCEPT-MANAGER: timp = 2
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 3
      AGENT left-turn-maneuver prohibited
PERCEPT-MANAGER: timp = 4
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 5
      AGENT left-turn-maneuver prohibited
PERCEPT-MANAGER: timp = 6
      AGENT left-turn-maneuver prohibited
```

Faptul că agentul poate vira la stânga este marcat prin valoarea "allowed" a manevrei left-turn-maneuver.
Iar in caz contrar manevra este marcată prin valoarea "prohibited".
Fiecare decizie luată a fost explicată mai sus.

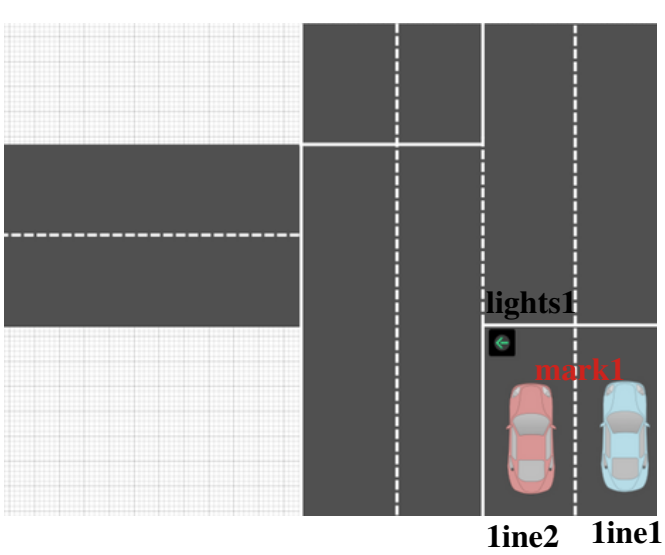
Raport de performanță

S-a rulat aplicatia de 5 ori și s-a înregistrat timpul de start si timpul de sfarșit, iar rezultatele au fost:

TIMP	MEDIA	SIGMA
T1	0,000081469319153	0,00003599169280993831
T2	0,000048398971557	0,000011400547233895569
T3	0,000080204010009	0,00002392254527455895
T4	0,000199794769287	0,0003085275058046078
T5	0,000071620941277	0,00001690974420591105
T6	0,000033535562284	0,00000755585438865924

- Media aritmetică a timpilor de execuție si deviatia standard este calculata in secunde (sec)

2.3 Viraj la stânga în funcție de încadrarea pe bandă și marcajul de pe drum



Descriere scenariu

Scenariu de față prezintă o autostradă cu 2 benzi pe ambele părți, și un drum la stânga, față de mașina autonomă, un drum cu 2 sensuri. Pentru a valida virajul la stânga este prezent un semafor a cărui culoare indică dacă poți face sau nu virajul la stânga. Masina autonomă este reprezentată de vehiculul albastru, care la diferite momente de timp se află pe banda 1 sau banda 2, și doreste sa ia virajul la stânga pe road1. În funcție de marcajul dintre cele 2 benzi si culoarea semaforului se va determina dacă agentul poate face manevra.

Percepții

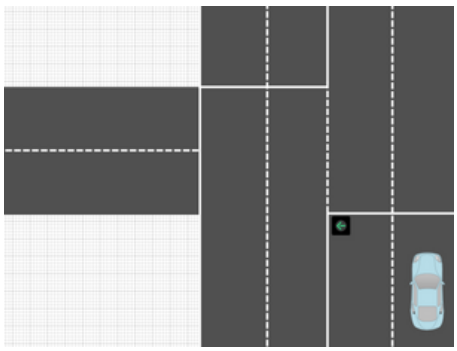
OBEJCT	NAME	VALUE
ev3	isa	eveniment
line1/2	isa	line
line1/2	partof	ev3
line1/2	status	on/free/busy
lights1	isa	lights
lights1	partof	ev3
lights1	color	green/red
lights1	direction	ahead
mark1	isa	mark1
mark1	partof	ev3
mark1	type	discontinuous/continuous

Descriere percepții

Valorile folosite pentru percepțiile de tip isa sunt eveniment,line, lights, mark aceastea indica tipul obiectelor care pot fi observate de masina autonomă. Percepțiile de tip partof au fost folosite pentru a indica apartenența obiectelor observate la un eveniment. Celelalte percepții sunt:

- direction - poate avea valorile ahead și este folosită pentru a indica direcția pe care se află semaforul relativ la masina autonomă.
- status - poate avea valorile on/ free/busy și este folosită pentru a indica faptul ca pe banda respectiva se afla masina autonoma,se află o altă masina, sau banda este liberă.
- color - indica culoarea semaforului, poate fi red sau green.
- type - poate fi continous sau discontiunous si indica daca marcajul este continu sau discontinuu, astfel încât dacă masina autonomă poate trece de pe o banda pe alta.

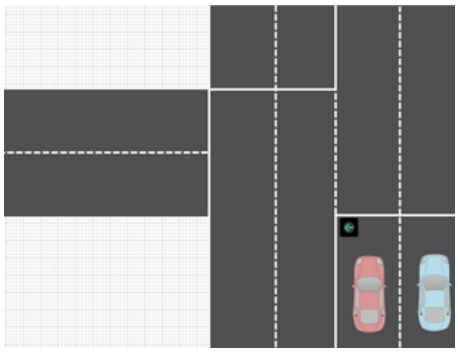
T1



Metrica

Agentul auto va lua decizia de a vira la stânga pentru că semaforul este verde, pe banda 2 nu se află o altă masină iar marcajul este unul discontinuu, astfel poate trece de pe o banda pe alta.

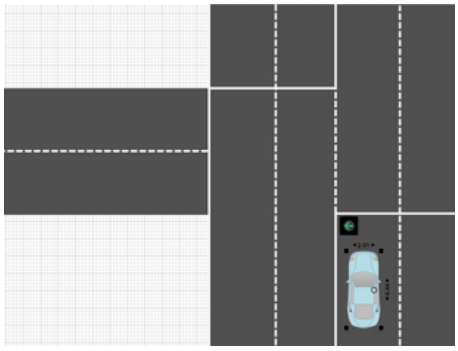
T2



Metrica

Agentul auto nu va putea vira la stânga deoarece se afla pe banda 1 iar pe banda 2 se află o alta masină care urmează sa facă stânga pe drum, astfel agentul auto nu se poate incadra pentru a efectua manevra.

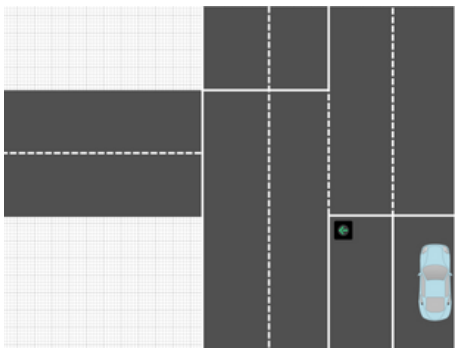
T3



Metrica

Semaforul este verde, agentul este încadrat pe banda pentru stânga, deci manevra de virare la stânga se poate realiza fără nici o problemă.

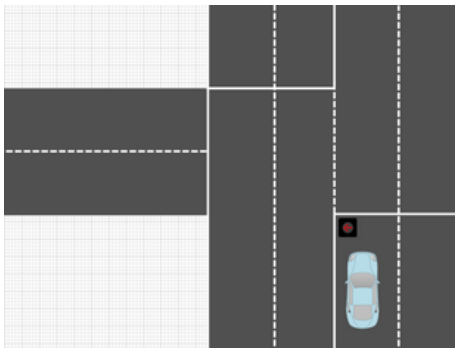
T4



Metrica

Deși semaforul este verde, iar pe banda 2 nu se află nici o mașina, agentul auto nu poate valida manevra de viraj la stânga deoarece marcajul este unul continuu, deci nu poate să se încadreze pe banda 2.

T5



Metrica

Indiferent pe ce bandă ar fi poziționat agentul auto, semaforul indică culoare roșie, deci virajul la stânga nu e posibil.

Output

```
PERCEPT-MANAGER: timp = 1
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 2
      AGENT left-turn-maneuver prohibited
PERCEPT-MANAGER: timp = 3
      AGENT left-turn-maneuver allowed
PERCEPT-MANAGER: timp = 4
      AGENT left-turn-maneuver prohibited
PERCEPT-MANAGER: timp = 5
      AGENT left-turn-maneuver prohibited
```

Faptul că agentul poate vira la stânga este marcat prin valoarea "allowed" a manevrei left-turn-maneuver.
Iar in caz contrar manevra este marcată prin valoarea "prohibited".
Fiecare decizie luată a fost explicată mai sus.

Raport de performanță

S-a rulat aplicatia de 5 ori și s-a înregistrat timpul de start si timpul de sfarşit, iar rezultatele au fost:

TIMP	MEDIA	SIGMA
T1	0,000073814392089	0,000012112039274999828
T2	0,000063800811767	0,000008504227290912486
T3	0,000040578842163	0,000008611829572742199
T4	0,000057792663574	0,000011410016749166528
T5	0,000044631958007	0,000013619524383150024

- Media aritmetică a timpilor de execuție si deviatia standard este calculata in secunde (sec)

3. Implementare

3.1 maneuverValidityASK.clp

Aici s-a înregistrat manevra care urmează să fie validată sau nu.

```
(deffacts AGENT::maneuvers-to-validate
    (ASK left-turn-maneuver)
)
```

3.2 DRIVER-AGENT.clp

Fisierul DRIVER-AGENT conține regulile impuse asupra agentului si decizile pe care acesta le face pe baza respectarii/nerespectarii regulilor. Fiecare manevra este compusa din 2 regulir generale: initCycle și Invalidate.

- **initCycle**

Caută un fapt slot existent în cod. Am ales să setez manevra de viraj la stânga ca allowed by default, deoarece pentru scenarile implementate de mine a fost mai usor să scriu reguli pentru nevalidarea manevrei decât pentru validarea acesteia. După procesarea invalidării pe fiecare scenariu, se v-a decide dacă manevrea este prohibited sau allowed.

```
(defrule AGENT::initCycle-left-turn
   (declare (salience 89))
   (timp (valoare ?)) ;make sure it fires each cycle
=>
   (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>initCycle-left-turn allowed by default "
   crlf))
   (assert (ag_bel (bel_type moment)  (bel_pname left-turn-maneuver) (bel_pval allowed))) ;by
   default, we assume left turn valid
   ;(facts AGENT)
)
```

- **Invalide**

Conține condițiilece se vor verifica pentru a decide dacă manevra poate fi realizată sau nu pentru fiecare scenariu având în vedere percepțiile din acesta.

Scenariu 1

```
(defrule AGENT::invalidare-scenariu1
(declare (salience -10))
(timp (valoare ?t))
(ag_percept (percept_pobj ev1) (percept_pname isa) (percept_pval event))
?f<-(ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval allowed))
(ag_bel (bel_type moment) (bel_pname trecere_pietoni1) (bel_pval yes))
(not  (ag_bel (bel_type moment) (bel_pname trecere_pietoni2) (bel_pval yes)))
(ag_percept (percept_pobj car1) (percept_pname intention) (percept_pval left I ahead ))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>invalidare-scenariu-1 " crlf))
  (retract ?f)
 (assert (ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval prohibited)))
)
```

Pentru a detecta trecerile de pietoni din stânga și din dreapta am implementat belief-uri de tip moment, care vor fi sterse din memorie la fiecare moment nou de timp.

```
(defrule AGENT::detectie-trecere-pietoni-dreapta
(timp (valoare ?t))
(ag_percept (percept_pobj ev1) (percept_pname isa) (percept_pval event))
(ag_percept (percept_pobj crossing2) (percept_pname direction) (percept_pval right))
(ag_percept (percept_pobj crossing2) (percept_pname state) (percept_pval busy))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-trecere-pietoni-dreapta " crlf))
    (assert (ag_bel (bel_type moment)   (bel_pname trecere_pietoni2) (bel_pval yes)))
    ;(facts AGENT)
)

(defrule AGENT::detectie-trecere-pietoni-stanga
(timp (valoare ?t))
(ag_percept (percept_pobj ev1) (percept_pname isa) (percept_pval event))
(ag_percept (percept_pobj ev1) (percept_pname isa) (percept_pval event))
(ag_percept (percept_pobj crossing1) (percept_pname direction) (percept_pval left))
(ag_percept (percept_pobj crossing1) (percept_pname state) (percept_pval busy))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-trecere-pietoni-stanga " crlf))
    (assert (ag_bel (bel_type moment)  (bel_pname trecere_pietoni1) (bel_pval yes)))
  ;  (facts AGENT)
)
```

Scenariu 2

```
(defrule AGENT::invalidare-scenariu2
(declare (salience -10))
(timp (valoare ?t))
?f<-(ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval allowed))
(or
(ag_bel (bel_type moment)  (bel_pname detectie-parcare) (bel_pval yes))
(ag_bel (bel_type moment)  (bel_pname detectie-intersectie) (bel_pval yes))
)
=>
 (retract ?f)
  (assert (ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval prohibited))
)
```

Pentru a detecta dacă culoarea semaforului este rosu am implementat un belief de tip fluent, care va fi sters din memorie în momentul în care se detectează că culoarea semaforului este verde.

```
(defrule AGENT::semafor-verde
(declare (salience 10))
(timp (valoare ?t))
?f<- (ag_bel (bel_type fluent)  (bel_pname semafor-rosu) (bel_pval yes))
(ag_percept (percept_pobj ev2 | ev3) (percept_pname isa) (percept_pval event))
(ag_percept (percept_pobj lights1) (percept_pname color) (percept_pval green))

=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-semafor-verde" crlf))
 (retract ?f)
  ; (facts AGENT)
)
```

```
(defrule AGENT::semafor-rosu
(declare (salience 10))
(timp (valoare ?t))
(ag_percept (percept_pobj ev2 | ev3) (percept_pname isa) (percept_pval event))
?fec<-(ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval allowed))
(ag_percept (percept_pobj lights1) (percept_pname color) (percept_pval red))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-semafor-rosu" crlf))
    (assert (ag_bel (bel_type fluent)  (bel_pname semafor-rosu) (bel_pval yes)))
 (retract ?fec)
  (assert (ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval prohibited)))
)
```

Pentru a semnaliza că există la momentul curent o masină în intersecție sau în parcare am implementat belief-uri de tip moment.

```
(defrule AGENT::detect-car-in-intersection
(not(ag_bel (bel_type fluent)  (bel_pname semafor-rosu) (bel_pval yes)))
(ag_percept (percept_pobj road1) (percept_pname direction) (percept_pval left))
(ag_percept (percept_pobj lane1) (percept_pname partof) (percept_pval road1))
(ag_percept (percept_pobj lane1) (percept_pname width) (percept_pval ?width_line))
(ag_percept (percept_pobj intersection1) (percept_pname direction) (percept_pval ahead))
(ag_percept (percept_pobj intersection1) (percept_pname heigth) (percept_pval ?heigth_intersection))

(ag_percept (percept_pobj car1) (percept_pname on) (percept_pval  intersection))
(ag_percept (percept_pobj car1) (percept_pname intention) (percept_pval  left))
(ag_percept (percept_pobj car1) (percept_pname heigth) (percept_pval ?heigth_car))
(ag_percept (percept_pobj car1) (percept_pname distantion_from_lights1) (percept_pval ?distance_from_lights ))
 (or
       (test ( < ( -  ?heigth_intersection ?heigth_car) ?width_line ))
       (test ( > ?distance_from_lights 1)))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-masina-in-intersectie" crlf))
  (assert (ag_bel (bel_type moment)  (bel_pname detectie-intersectie) (bel_pval yes)))
)
```

```
(defrule AGENT::detect-car-in-parking
(not(ag_bel (bel_type fluent)  (bel_pname semafor-rosu) (bel_pval yes)))
(ag_percept (percept_pobj lights1) (percept_pname proximity) (percept_pval ?proximity_lights))
(ag_percept (percept_pobj parking1) (percept_pname proximity) (percept_pval ?proximity_parking ))
(ag_percept (percept_pobj car1) (percept_pname on) (percept_pval  parking1))
(ag_percept (percept_pobj car1) (percept_pname intention) (percept_pval  back))
(test (> ?proximity_parking ?proximity_lights))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-masina-in-parcare" crlf))
  (assert (ag_bel (bel_type moment)  (bel_pname detectie-parcare) (bel_pval yes)))
)
```

Scenariu 3

```
(defrule AGENT::invalidare-scenariu3
(declare (salience -10))
(timp (valoare ?t))
?f<-(ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval allowed))
(or
(ag_bel (bel_type moment)   (bel_pname linie-continua)  (bel_pval yes))
(and
(ag_bel (bel_type moment)   (bel_pname linie-discontinua) (bel_pval yes))
(ag_percept (percept_pobj line2) (percept_pname status) (percept_pval busy))))
=>
 (retract ?f)
  (assert (ag_bel (bel_type moment) (bel_pname left-turn-maneuver) (bel_pval prohibited)))
)
```

Pentru a vedea tipul liniei la orice moment de timp am implementat belif-uri de tip moment care să determine acest lucru.

```
(defrule AGENT::marcaj-linie
(timp (valoare ?t))
(ag_percept (percept_pobj ev3) (percept_pname isa) (percept_pval event))
(ag_percept (percept_pobj line1) (percept_pname status) (percept_pval on))
(ag_percept (percept_pobj mark1) (percept_pname type) (percept_pval ?value))
=>
 (if (eq ?*ag-in-debug* TRUE) then (printout t "    <D>detectie-marcaj-linie " crlf))
 (if (eq ?value  continuous)  then     (assert (ag_bel (bel_type moment)   (bel_pname linie-continua) (bel_pval yes)))    )
  (if (eq ?value  discontinuous)  then         (assert (ag_bel (bel_type moment)   (bel_pname linie-discontinua) (bel_pval yes)))   )
  ;  (facts AGENT)
)
```

Pentru a observa culoarea semaforului la un moment de timp, m-am folosit de belfiuri-le de tip fluent implementate pentru scenariu 2.

4. Obținerea percepțiilor

Percepțiile sunt colectate în mai multe moduri folosind diferiti senzori (vizuali, sonori, distanță) sau prin softaware-uri de procesare de imagini.
Masina autonoma este dotată cu camere de vedere montate în direcția parbrizului și deasupra masini astfel tot mediul înconjurator este observat. Cu ajutorul camerelor 3D imaginile detectate sunt foarte realistice și detaliate. Aceste camere conțin senzori care detectează automat obiectele din jur (alte masini, pietoni, semne de circulație) , le clasifică în funcție de categori și determină distanța dintre ele și autovehicul. Totuși există unele slabiciuni, cum ar fi în cazul unor conditii meteo nefavorabile
(zapadă, ploaie, ceață), camerele nu pot detecta clar obstacolele a căror culori sunt asemânătoare.
Un conducător auto prin intermediul oglinzilor are posibilitatea doar de a vedea dacă exisra un vehicul auto în spatele lui, prin aceste camere se poate observa o distanță mai mare în spatele lui.
Cu ajutorul acestor senzori pe board-ul de navigație vor apărea semnele de circulație în timp real, ba chiar preventiv pentru a putea lua decizi optime în prealabil.
Acuratețea GPS-ului este foarte importantă pentru o percepție corecta, deoarece este nevoie de informații exacte, fără greseli. GPS-ul este preîncărcat cu hârți și semnele de circulație ale acestora, deorece în cazul în care o detecție a camerei dă un răspuns fals pozitiv (după cum am menționat mai sus), agentul să se bazeze pe date predefinite.
Senzorii radar au de asemenea o contribuție crucială la funcția de conducere autonomă. Ei trimit unde radio care detectează obiecte și le măsoară distanța și viteza în raport cu masina autonomă, în timp real. Există 2 tipuri de senzori: cu rază scurtă care monitorizează punctul mort, menținerea benzii și oferă ajutor la parcare; cu rază lungă asigură distanța dintre diferite obiecte și oferă asistență la frânare.
Spre deosebire de senzori camerei, sistemele radar nu au probleme atunci când identifică obiecte în timpul ceții sau ploii. Senzori radar scanează doar pe orizontală, ceea ce poate cauza o varietate de probleme atunci când urmează să conduci sub un pod.
O masină autonoma prezintă și senzori Lidar, care funcționează asemânatori cu cei radar, singura diferență fiind că folosesc lasere în loc de unde radio. Acesti senzori crează o hartă de 360 grade în jurul autovehicului.
Deep Learning-ul este baza masinilor autonome. Pentru clasificarea tipurilor obiectelor detectate se va folosi un algoritm de procesare a imaginilor (object detection) . Algoritmul de recunoastere a pietonilor necesită îmbunătățirii deoarece vehiculul auto identifică corect numai 90%-95% din pietoni .

Percepțiile se găsesc în folderele scenariu1/2/3.

1. (ag_percept (percept_probj crossing1) (percept_pname isa) (percept_pval ped_crossing))
   (ag_percept (percept_probj crossing1) (percept_pname state) (percept_pval free/busy))

Detecția treceri de pietoni se face prin intermediul GPS-ul care după cum am specificat mai sus conține harta drumului cu toate semnele de circulație. Aceasta detecție este confirmată și de senzorii camerei montate pe masină. Camera foloste algoritmi de procesare a imaginilor (object detention/ body detection) cu ajutorul cărora detectează marcajele albe de pe drum și miscarea oamenilor deasupra acesteia. În cazul în care nu apar persoane în cadrul marcajului aceasta este free altfel este busy. Algoritmi sunt implementați prin openCV și python.

- https://data-flair.training/blogs/pedestrian-detection-python-opencv/
- https://answers.opencv.org/question/226261/way-to-find-zebra-crossing-lines/

2. (ag_percept (percept_probj sign1) (percept_pname isa) (percept_pval road_sign))
   (ag_percept (percept_probj sign1) (percept_pname type) (percept_pval priority))

GPS-ul este actualizat cu toate semnele de circulație, astfel în momentul în care camera dectează un nou semn de circulație acesta este confirmat și de prezența lui pe GPS-ului. Se folosesc tot algoritmi de detecție a obiectelor. Se detectează semnele de circulație în formă de romb după care se verifică dacă culoarea din interior este galbenă. GPS-UL are o acuratete de 3 metri conform celui de al doilea link de mai jos. Pozitia data de GPS poate fi eronata si din cauza faptului ca semnalul GPS de la satelit poate fi blocat de cladiri.

- https://github.com/harsh-99/Traffic-sign-detection
- https://www.brickhousesecurity.com/gps-trackers/gps-accuracy/

3. (ag_percept (percept_probj road1) (percept_pname isa) (percept_pval road))
   (ag_percept (percept_probj lane1) (percept_pname isa) (percept_pval line))
   (ag_percept (percept_probj line1) (percept_pname status) (percept_pval on/busy/free))

Detecția unui drum/benzi se face cu ajutorul GPS-ului momentat pe masină. Aici este un top a celor mai bune gps-uri de pe piață la momentul actual. Cu ajutorul camerei de luat vederi și a unei procesari de imagini auxiliare se poate indentifica dacă există o masina pe o anumita banda/drum. Iar faptul că masina mea se afla pe o anumită bandă/drum este dat de localizarea de pe GPS. In ultimul link se poate observa acuratetea unui algoritm de detectie a unei benzi, aceasta acuratetea depinde de mai multi factoric care sunt specificati la link-ul respectiv.

- https://www.toptenreviews.com/best-car-navigation-system
- https://medium.com/mlearning-ai/road-lane-detection-using-opencv-hough-lines-transform-explained-a6c8cfc03f68
- https://cv.utcluj.ro/index.php/scabor.html

Road Type	Traffic	Avg. Detection Rate Per Minute		
		Correct	Incorrect	Misses
Isolated	Light	99.08%	0.99%	0%
Highway	Moderate	98.34%	1.65%	0%
Metro	Light	98.37%	1.65%	0%
Highway	Moderate	96.34%	3.65%	0%
City	Variable	86.39%	12.71%	0.78%

4. (ag_percept (percept_probj car1) (percept_pname isa) (percept_pval car))
   (ag_percept (percept_probj car1) (percept_pname on) (percept_pval road1/parking1))
   (ag_percept (percept_probj car1) (percept_pname intention) (percept_pval right/ahead/left/none/back))

Aceasta percepție se poate obține cu ajutorul unei camere montate pe mașină care procesează imaginile în timp real și determina forma acesteia ca fiind o altă masină, totodată îi determina poziția. Putem observa o mașină că semnalizează tot cu ajutorul unei camere care preia imagini în timp real, la un interval suficient de scurt de timp, astfel încat să își dea seama că culoarea becurilor de semnalizare se schimbă întermitent și care dintre cele 2 semnalizari se schimbă.

- https://kalebujordan.dev/real-time-vehicle-detection-using-python/

5. (ag_percept (percept_probj parking1) (percept_pname isa) (percept_pval parking))

Locul de parcare este detectat prin intermediul camerei și a GPS-ului, dar și prin intermediul senzorilor Lidar. Despre acuratetea senzorului Lidar se poate accesa linkul al doilea.

- https://link.springer.com/article/10.1007/s13177-022-00300-w
- https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1902&context=open_access_theses

6. (ag_percept (percept_probj lights1) (percept_pname isa) (percept_pval traffic_lights))
   (ag_percept (percept_probj lights1) (percept_pname color) (percept_pval green/red))

Detectarea semnelor de circulatie si a semafoartelor se folosesc librarii opencv si clasificatoare in cadrul camerelor de din fata masinii. Vehicolul este de asemenea echipat cu GPS cu harti pre-descarcate, asadar în situații de vizibilitate minima agentul este atenționat asupra semnelor. Sistemul de navigatie poate fi conectat la internet, asadar raimprospatarea harților este posibila.Pentru a dectecta ce culoare are semaforul la momentul curent se folosesc algoritmi implemtați în openCV care utilizează histograma pentru a determina culoarea:

- https://stackoverflow.com/questions/64723166/how-to-detect-trafic-light-color-with-histogram-info-using-opencv

7. (ag_percept (percept_probj intersection1) (percept_pname isa) (percept_pval intersection))

Camera montată în fața masini detectează prin algoritmi de procesare a imaginilor faptul ca urmează o intersecție, stiind ca marcajul de final de strada este alb, detectând linia alba.

- https://medium.com/mlearning-ai/road-lane-detection-using-opencv-hough-lines-transform-explained-a6c8cfc03f68

8. (ag_percept (percept_probj mark1) (percept_pname isa) (percept_pval mark))
   (ag_percept (percept_probj mark1) (percept_pname type) (percept_pval discontinuous/continous))

Detecția unui marcaj de pe drum se face cu ajutorul camerei montate deasupra masini și in față masini. Iar dupa ce imaginile au fost preluate se aplică algoritmi de detecție a tipului de linie pentru a afla daca este continua sau discontinua.

- https://ori.ox.ac.uk/media/5701/road-marking-classification-paul-ingmar.pdf

9. (ag_percept (percept_probj _____) (percept_pname proximity) (percept_pval 0/1))

Pentru a afla care obiect se afla mai aproape de autohevicul se va folosi radar-ul masini care calculeaza distanța de la un obiect la masină. După ce toate distanțele sunt calculate se pun intr-un sir în funcție de distanța cea mai mică.

- https://pyimagesearch.com/2016/04/04/measuring-distance-between-objects-in-an-image-with-opencv/

10 .(ag_percept (percept_probj _) (percept_pname width) (percept_pval _))
   (ag_percept (percept_probj car2) (percept_pname heigth) (percept_pval 3))

Imagini cu obiectele sunt preluate prin intermediul camerelor montate pe masină. Pe urmă, aceste imagini sunt procesate după cum se poate observa în link-ul de mai jos:

- https://pyimagesearch.com/2016/03/28/measuring-size-of-objects-in-an-image-with-opencv/

11. (ag_percept (percept_probj _) (percept_pname direction) (percept_pval left/right/ahead))

Direcția unui obiect față de mașină se dectează prin senzori camerei. Se preaia imaginea de pe camera din față după care se imparte în două jumatați egale iar apoi în funcție de origine se dectează de ce parte este (stânga sau dreaptă). Un exemplu de decupare după origine și aflarea de ce partea a origine se afla:

- https://stackoverflow.com/questions/59525065/detecting-object-location-in-image-with-python-opencv

5. Concluzii și limitări

In functie de percepțiile scrise în cadrul fiecarui scenariu, regula implementata va afisa ca output dacă virajul la stânga este permis sau nu și ce tip de scenariu este și va aserta in memorie faptul corespunzator.
Limitările programului ar fi în cazul scenariului 2 de exemplu pentru cazul T3, deși masina încalcă regula 1, s-ar putea să se afle la o distanță mai mica decât 1 față de semafor, altfel încât să treacă de linia de simetrie a drumului, și agentul auto să aibe spațiu pentru a vira, dar aceasta situație nu am implementat-o.