Design of a Semplified Automous Driving System for Traffic Light

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

Department of Industrial Engineering - University of Trento

November 22, 2022

Objective

Course objective

Implementation of longitudinal control of an autonomous agent that manages traffic lights and intersections

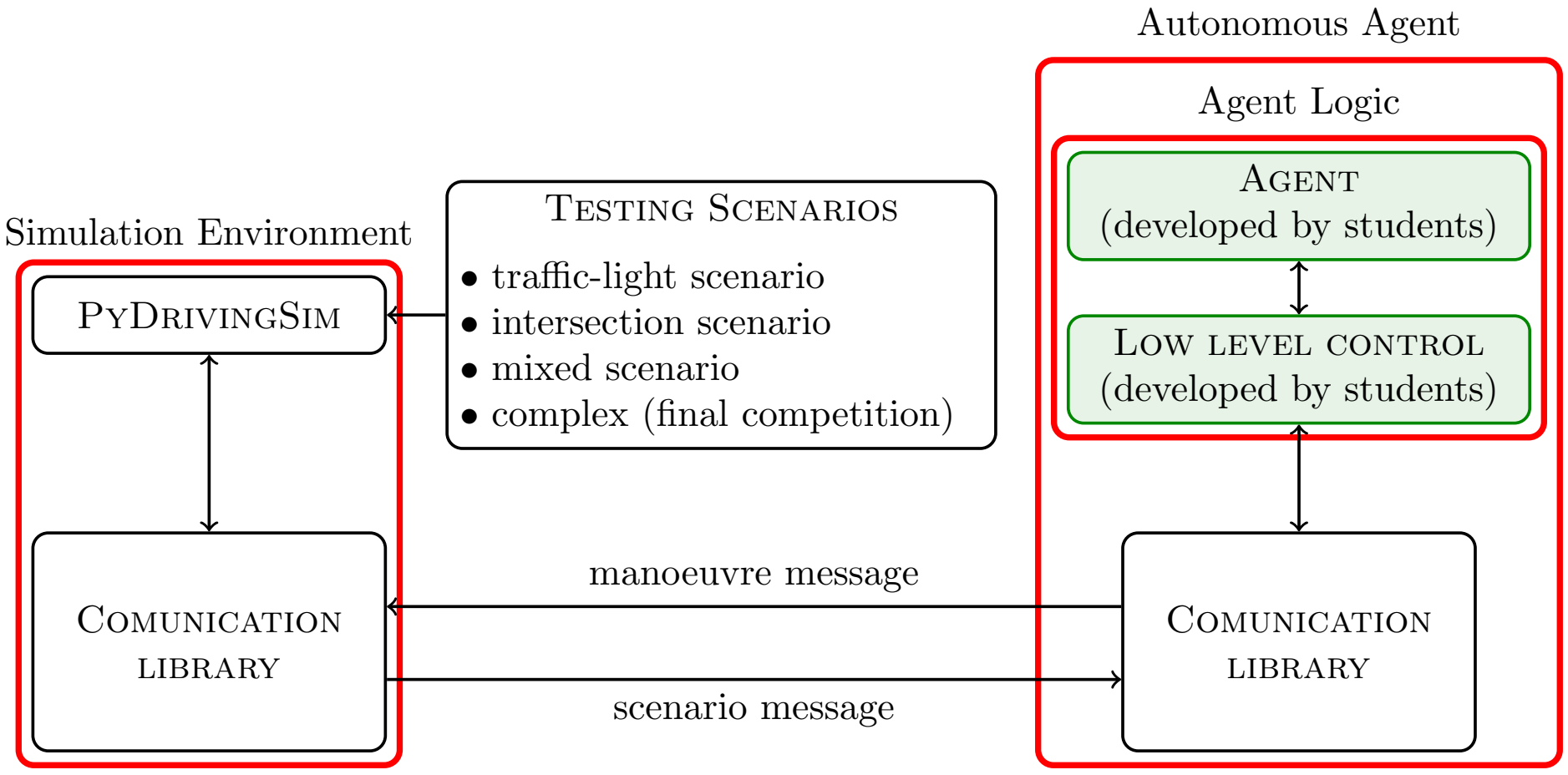
---

Mauro Da Lio, Alessandro Mazzalai, and Marco Darin. "Cooperative Intersection Support System Based on Mirroring Mechanisms Enacted by Bio-Inspired Layered Control Architecture". In: *IEEE Transactions on Intelligent Transportation Systems* 19.5 (2018).

Implementation

- Implementation: any language (MATLAB, C++)
- Communication: UDP connection by dynamic library
- Testing environment: PyDrivingSim

Agent Implementation and Testing



Autonomous Agent

Layered architecture by Brooks and human like behaviours

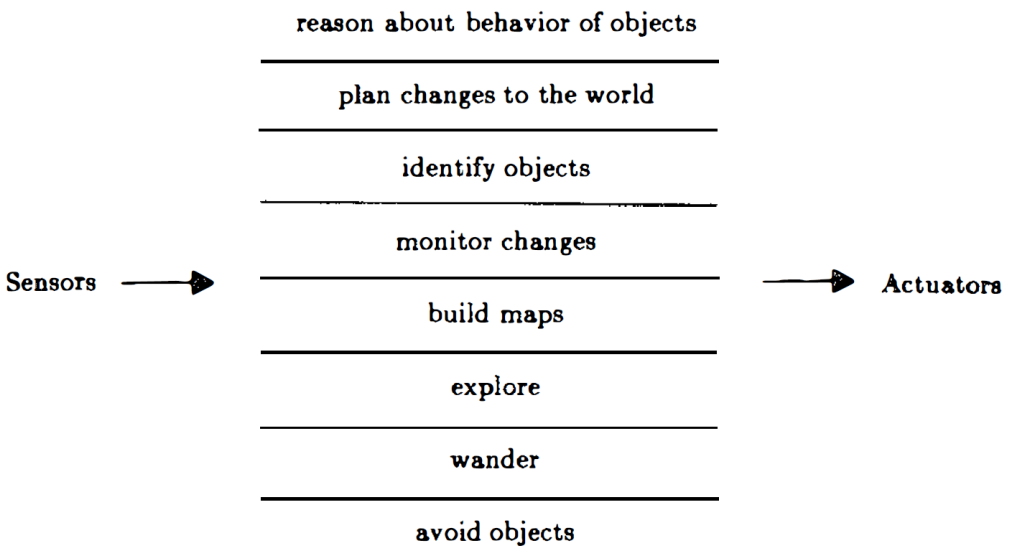


Fig. 2. Decomposition of a mobile robot control system based on task-achieving behaviors.

Humanlike sensorimotor behaviours

- Action Priming (generation of affordances): based minimal intervention principle
- Action Selection: based on basal ganglia structure

Subsumption architectures

Rodney A Brooks. "A robust layered control system". In: *IEEE Journal of Robotics and Automation* 2.I (1986).

Autonomous Agent

Action Priming (Generation of affordances)

Based on simple pant[a], and minimum square jerk criterion[b]

$$\begin{cases} \dot{s}(t) = v(t) \\ \dot{v}(t) = a(t) \\ \dot{a}(t) = j(t) \end{cases} \qquad J = \int_0^T j(t)^2 dt$$

$$s(t) = c_1 t + \frac{1}{2} c_2 t^2, + \frac{1}{6} c_3 t^3 + \frac{1}{24} c_4 t^4 + \frac{1}{120} c_5 t^5$$

---

[a] Mauro Da Lio et al. "Biologically guided driver modeling: The stop behavior of human car drivers". In: *IEEE Transactions on Intelligent Transportation Systems* 19.8 (2018).

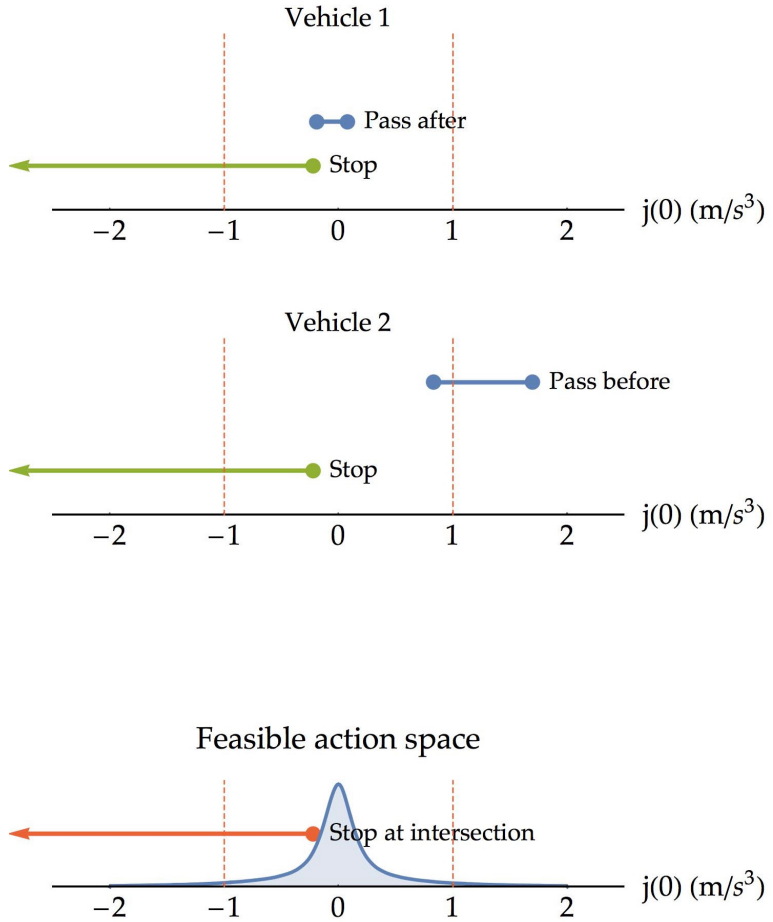
[b] Emanuel Todorov and Michael I Jordan. "A minimal intervention principle for coordinated movement". In: *Advances in neural information processing systems* 15 (2003).

Autonomous Agent

Action Selection

The agent take inspiration from basal gaglia structure

- Action representation
- Inhibition mechanism
- Mirroring



Rafal Bogacz and Kevin Gurney. "The basal ganglia implement optimal decision making between alternative actions". In: *Review Literature And Arts Of The Americas* (2005).

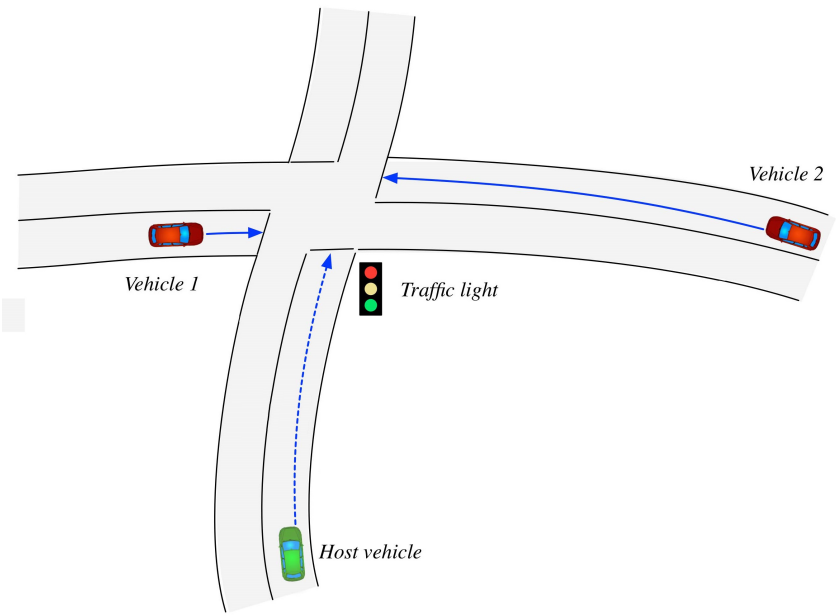
Communication Library

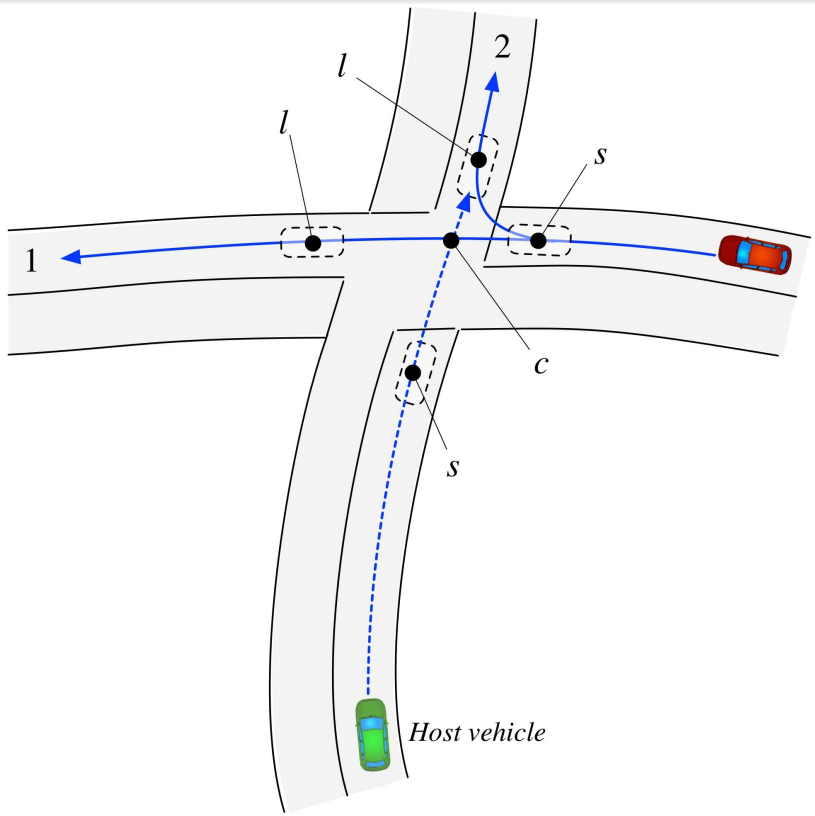
```
// Fuction for testing library loading
void test_lib();
// Server Init Matlab
void server_agent_init_num(unsigned int num_ip[4], int
    server_port);
// Server agent Init
void server_agent_init(const char *server_ip, int server_port);
// Server agent Send
int server_send_to_client(uint32_t server_run, uint32_t
    message_id, output_data_str* manoeuvre_msg);
// Server agent Receive
int server_receive_from_client(uint32_t *server_run, uint32_t *
    message_id, input_data_str* scenario_msg);
// Server get ip of the client
char* server_receive_from();
// Server agent Close
void server_agent_close();
```

Simulation Environment

Application Scenario

- Traffic light
- Intersecting lanes





Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Agent Definition

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

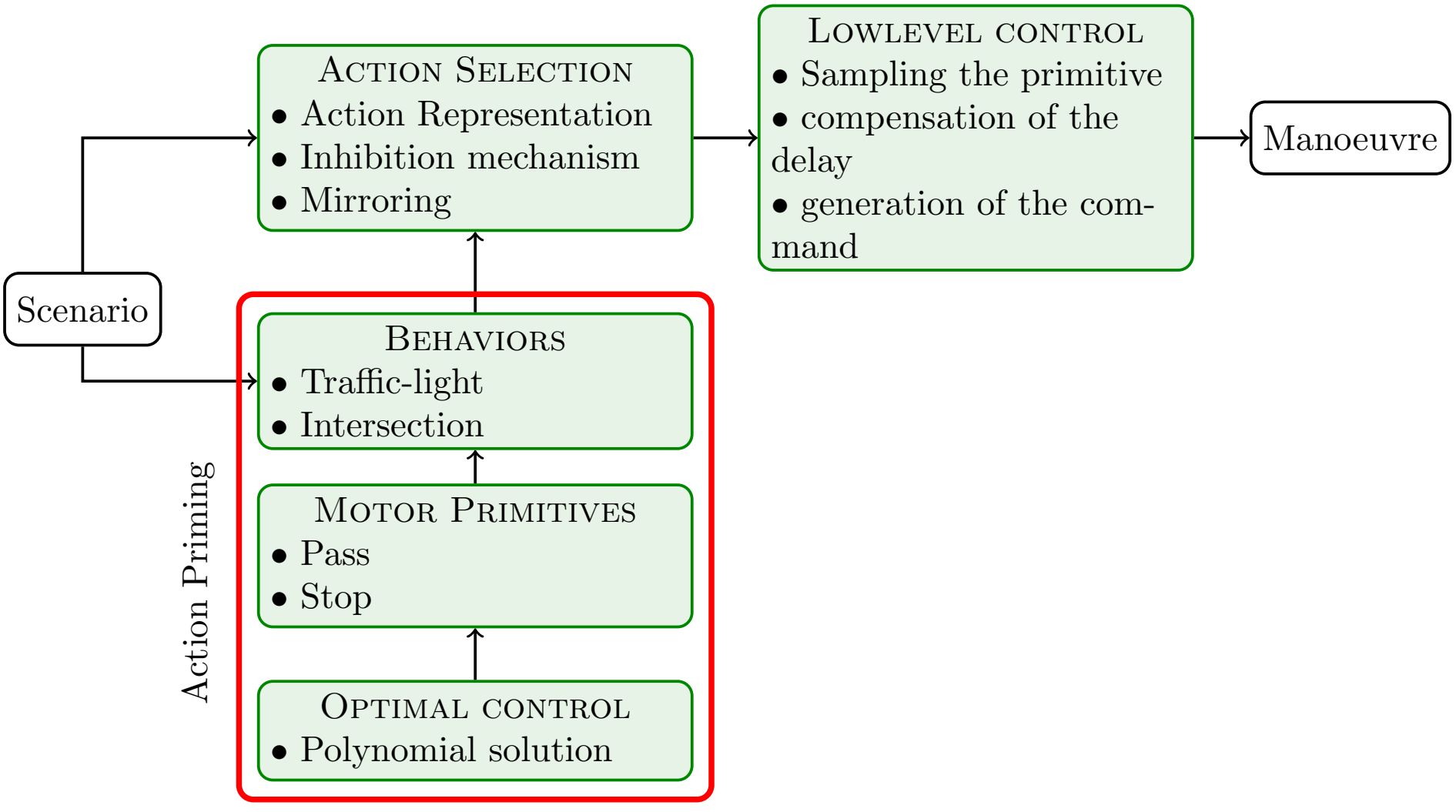
Department of Industrial Engineering - University of Trento

November 22, 2022

Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Agent structure



Developing Approach the Autonomous Agent

Top down approach

- Useful to understand the main component of the agent

Bottom up approch

- Necessary to developing the agent from the basic functionality to the higher motor cognitive abilities

Simple agent structure - Initialization of the server

```
int main(int argc, const char * argv[]) { ...
// enable log
logger.enable(true);

// Create messages variables
scenario_msg_t  scenario_msg;
manoeuvre_msg_t manoeuvre_msg;

// Initialization of the comunication
server_agent_init(DEFAULT_SERVER_IP, SERVER_PORT);

// Start server
while(server_run == 1) {
   ...
```

Simple agent structure - Receiving the message and log

```
// New message from the environment is received
if (server_receive_from_client(&server_run, &message_id, &
  scenario_msg.data_struct) == 0) { ...

  // Get the informations from the scenario msg
  input_data_str *in = &scenario_msg.data_struct;

  // Log a varible
  // logger.log_var("FILE", "VAR_NAME", VAR_VALUE);
  logger.log_var("Example", "cycle", in->CycleNumber);
  // Write log
  logger.write_line("Example");

  ...
```

Simple agent structure - computing the manoeuvre

```
...

// Is computed the manoeuvres using the scenario variables
manoeuvresList = actionPriming(in);
// For the intersection (other cars) the manoeuvres are
computed

// The best action based on the scenario is defined
bestManoeuvre = actionSelection(...);
// Using the inhibition mechanism and the mirroring the
best action is chosen}

...
```

Simple agent structure - generating the command

```
    ...

    // The low level control is used to generate the requested
    acceleration
    RequestedAcc = LowLevelControl(...);

    // The manoeuvre is sent to the environment
    if (server_send_to_client(server_run, message_id, &
    manoeuvre_msg.data_struct) == -1) {  ...
      // Restarting the cycle
    }
  }
}
```

Simple agent structure - closing the server

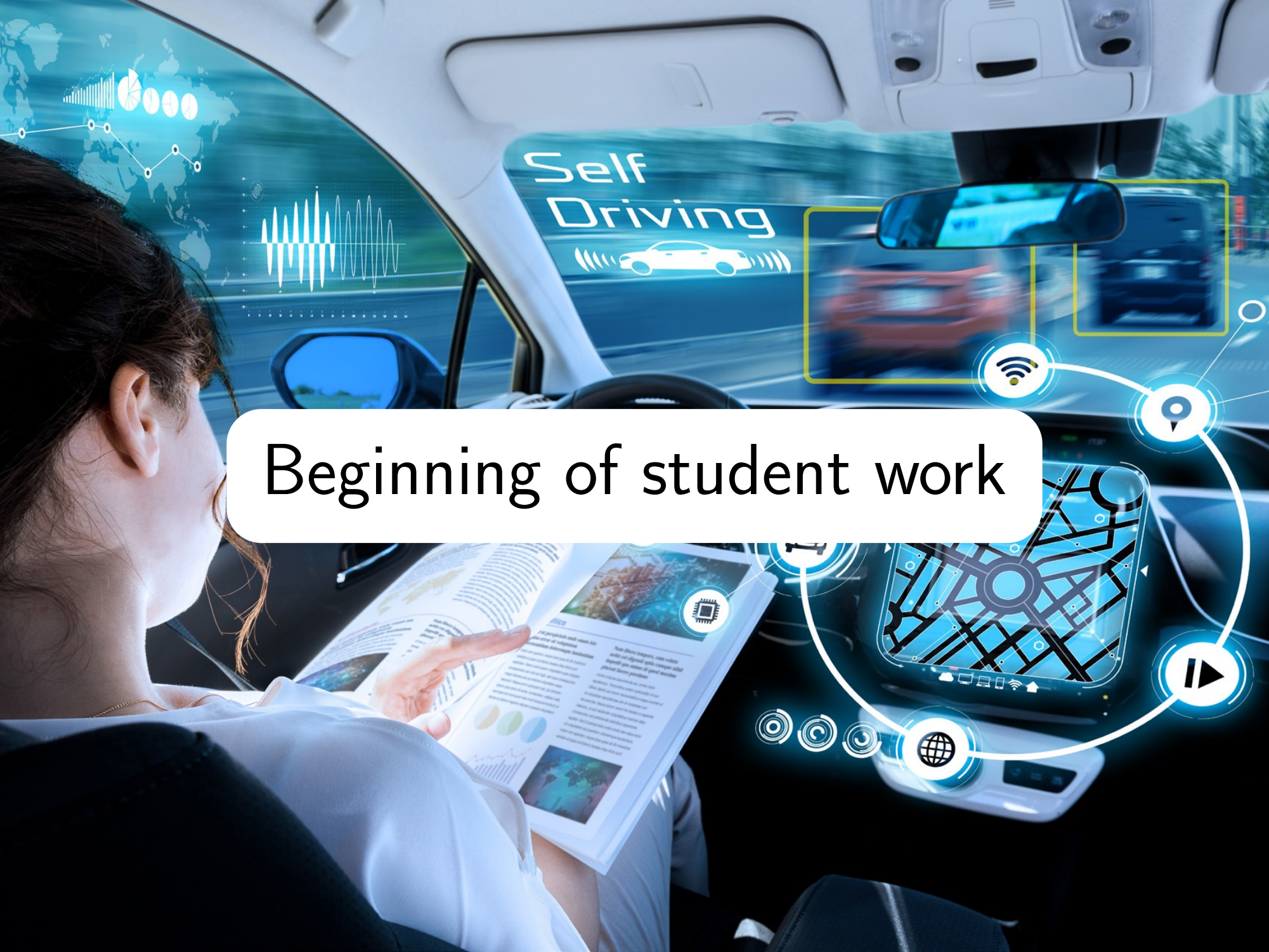
```
...

// Close the server of the agent
server_agent_close();
return 0;
}
```

Essential programs and files

Course prerequisites

- GIT (https://git-scm.com/)
- CLion clion (https://www.jetbrains.com/clion/download/) or Visual Studio
- Students materials:
  - Simple example https://bitbucket.org/tonegas/basic_agent_st/src/master/
  - Communication library
  - Simulation environment https://github.com/tonegas/PyDrivingSim



Beginning of student work

From the Minimum Jeck to the Motor Primitives

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

Department of Industrial Engineering - University of Trento

November 22, 2022

Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Optimal Control Recap - 1/3

System

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$$

Objective

Find an admissible control policy $\mathbf{u}(t)^*$ capable to steer the system from any $\mathbf{x}(0)$ (initial condition) to target state $\mathbf{x}(t_f)$ in a (free) time t_f while minimizing a cost function.

Optimal Control Recap - 2/3

Abstract framework for optimization problem

$$J\left(\mathbf{x}, \mathbf{u}, t_f, t_0\right) = \underbrace{\phi\left(\mathbf{x}\left(t_0\right), t_0, \mathbf{x}\left(t_f\right), t_f\right)}_{\text{Mayer}} + \int_{t_0}^{t_f} \underbrace{L(\mathbf{x}(t), \mathbf{u}(t), t)}_{\text{Lagrange}}\, dt$$

Where $L\left(\mathbf{x}(t), \mathbf{u}(t), t\right) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ provides a cost measure along the trajectory followed by the system for $t \in [t_0, t_f)$ with initial state $x(t_0)$ and $\phi\left(\mathbf{x}(t_0), t_0, \mathbf{x}(t_f), t_f\right) : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ represents the cost depending on the final state $\mathbf{x}(t_f)$.

Optimal control Recap - 3/3

Solving through the Hamiltonian

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}, t) = \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}, \mathbf{u}) + L(\mathbf{x}(t), \mathbf{u}(t), t)$$

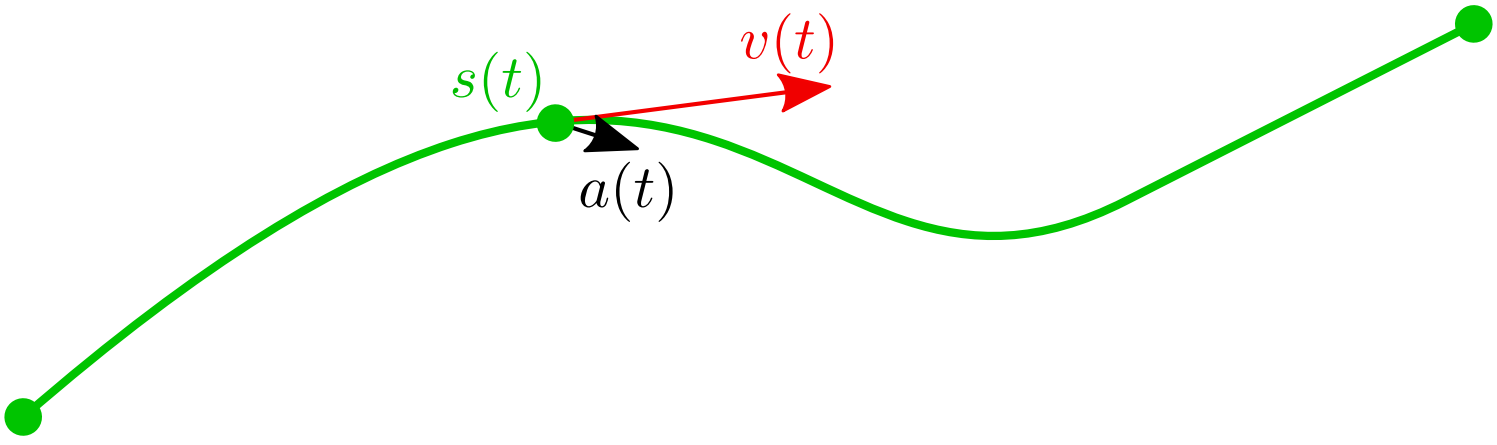
Optimality conditions for the Hamiltonian:

$$1) \quad \dot{\mathbf{x}} = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$$

$$2) \quad \dot{\boldsymbol{\lambda}} = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}$$

Hamiltonian minimization is equivalent to the Lagrangian minimization.

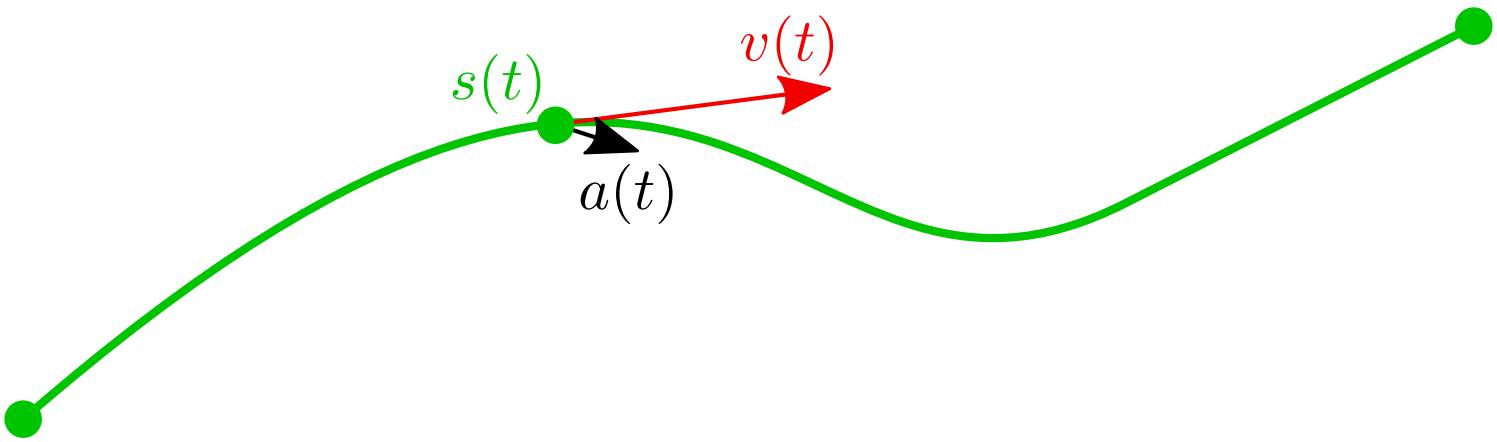
Longitudinal model - 1/2



Longitudinal kinematic vehicle model

$$\mathbf{x}(t) = \begin{pmatrix} s(t) \\ v(t) \\ a(t) \end{pmatrix}$$

Longitudinal model - 1/2



Longitudinal kinematic vehicle model

$$\dot{\mathbf{x}}(t) = \overbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}}^{A} \mathbf{x}(t) + \overbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}^{B} \overbrace{j(t)}^{\mathbf{u}(t)} \qquad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$$

Optimal control problem - 1/9

Minimization problem

$$\min_{j(t)} \int_0^{t_f} j(t)^2 dt$$

Lagrangian terms

$$L(j(t)) = j(t)^2$$

Lagrangian cost in our optimal control formulation. No Mayer term.

Optimal control problem - 2/9

Lagrangian Multipliers

$$\boldsymbol{\lambda}^T = \{\lambda_1(t), \lambda_2(t), \lambda_3(t)\}$$

Definition of the Hamiltonian

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}, t) = \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}, \mathbf{u}) + L(\mathbf{x}(t), \mathbf{u}(t), t)$$

Hamiltonian for longitudinal problem

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, j) = \overbrace{\lambda_1(t)v(t) + \lambda_2(t)a(t) + \lambda_3(t)j(t)}^{\boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x},\mathbf{u})} + j(t)^2$$

Hamiltonian minimization is equivalent to the Lagrangian minimization.

Optimal control problem - 3/9

Solving the Hamiltonian

$$\frac{\partial \mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, j(t))}{\partial j(t)} = 0$$

Deriving the Hamiltonian and imposing it equal to zero we obtain the minimization of the Hamiltonian

Solution of the optimal control problem

$$j(t) = -\frac{\lambda_3(t)}{2}$$

This control lead to the minimization

$$\int_0^{t_f} j(t)^2 dt$$

Optimal control problem - 4/9

Hamiltonian

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}, t) = \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}, \mathbf{u}) + L(\mathbf{x}(t), \mathbf{u}(t), t)$$

First optimality condition

$$\dot{\mathbf{x}} = \frac{\partial \mathcal{H}}{\partial \lambda} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \quad \Rightarrow \quad \dot{\mathbf{x}}(t) - \mathbf{f}(\mathbf{x}, \mathbf{u}) = 0$$

First constraint to the optimal control problem

$$\dot{s}(t) - v(t) = 0$$
$$\dot{v}(t) - a(t) = 0$$
$$\dot{a}(t) - j(t) = 0$$

Optimal control problem - 5/9

Hamiltonian

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}, t) = \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}, \mathbf{u}) + L(\mathbf{x}(t), \mathbf{u}(t), t)$$

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}, j) = \lambda_1(t)v(t) + \lambda_2(t)a(t) + \lambda_3(t)j(t) + j(t)^2$$

Second optimality condition

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}$$

Second constraint to the optimal control problem

$$\lambda_1'(t) = 0$$
$$\lambda_2'(t) + \lambda_1(t) = 0$$
$$\lambda_3'(t) + \lambda_2(t) = 0$$

Optimal control problem - 6/9

Boundary condition on initial state

$$s(0) = 0$$
$$v(0) = v_0$$
$$a(0) = a_0$$

Boundary condition on final state

$$s(t_f) = s_f$$
$$v(t_f) = v_f$$
$$a(t_f) = a_f$$

Optimal control problem - 7/9

We obtain a system of differential equations

Equations due to the system

$$\dot{s}(t) - v(t) = 0$$
$$\dot{v}(t) - a(t) = 0$$
$$\dot{a}(t) + \lambda_3(t)/2 = 0$$
$$\lambda_1'(t) = 0$$
$$\lambda_2'(t) + \lambda_1(t) = 0$$
$$\lambda_3'(t) + \lambda_2(t) = 0$$

Equations due to the initial conditions

$$s(0) = 0$$
$$v(0) = v_0$$
$$a(0) = a_0$$
$$s(t_f) = s_f$$
$$v(t_f) = v_f$$
$$a(t_f) = a_f$$

Optimal control problem - 8/9

Solution of the optimal control problem

$$s_{\text{opt}}(t) = c_1 t + \frac{1}{2}c_2 t^2 + \frac{1}{6}c_3 t^3 + \frac{1}{24}c_4 t^4 + \frac{1}{120}c_5 t^5$$

$$v_{\text{opt}}(t) = c_1 + c_2 t + \frac{1}{2}c_3 t^2 + \frac{1}{6}c_4 t^3 + \frac{1}{24}c_5 t^4$$

$$a_{\text{opt}}(t) = c_2 + c_3 t + \frac{1}{2}c_4 t^2 + \frac{1}{6}c_5 t^3$$

$$j_{\text{opt}}(t) = c_3 + c_4 t + \frac{1}{2}c_5 t^2$$

Optimal control problem - 9/9

Coefficients of the optimal solution

$$c_1 = v_0$$

$$c_2 = a_0$$

$$c_3 = \frac{3a_f - 9a_0}{t_f} + \frac{60s_f}{t_f^3} - \frac{12\left(2v_f + 3v_0\right)}{t_f^2}$$

$$c_4 = \frac{36a_0 - 24a_f}{t_f^2} - \frac{360s_f}{t_f^4} + \frac{24\left(7v_f + 8v_0\right)}{t_f^3}$$

$$c_5 = \frac{60\left(a_f - a_0\right)}{t_f^3} + \frac{720s_f}{t_f^5} - \frac{360\left(v_f + v_0\right)}{t_f^4}$$

Useful functions - 1/2

Definition

- we define the coefficients of the polynomial:

$$m = \{c_1, c_2, c_3, c_4, c_5\}$$

Useful functions

- for calculating the coefficients of the polynomial:

$$\text{evalPrimitiveCoeffs}(v_0, a_0, s_f, v_f, a_f, t_f) \Rightarrow \{c_1, c_2, c_3, c_4, c_5\}$$

Useful functions - 2/2

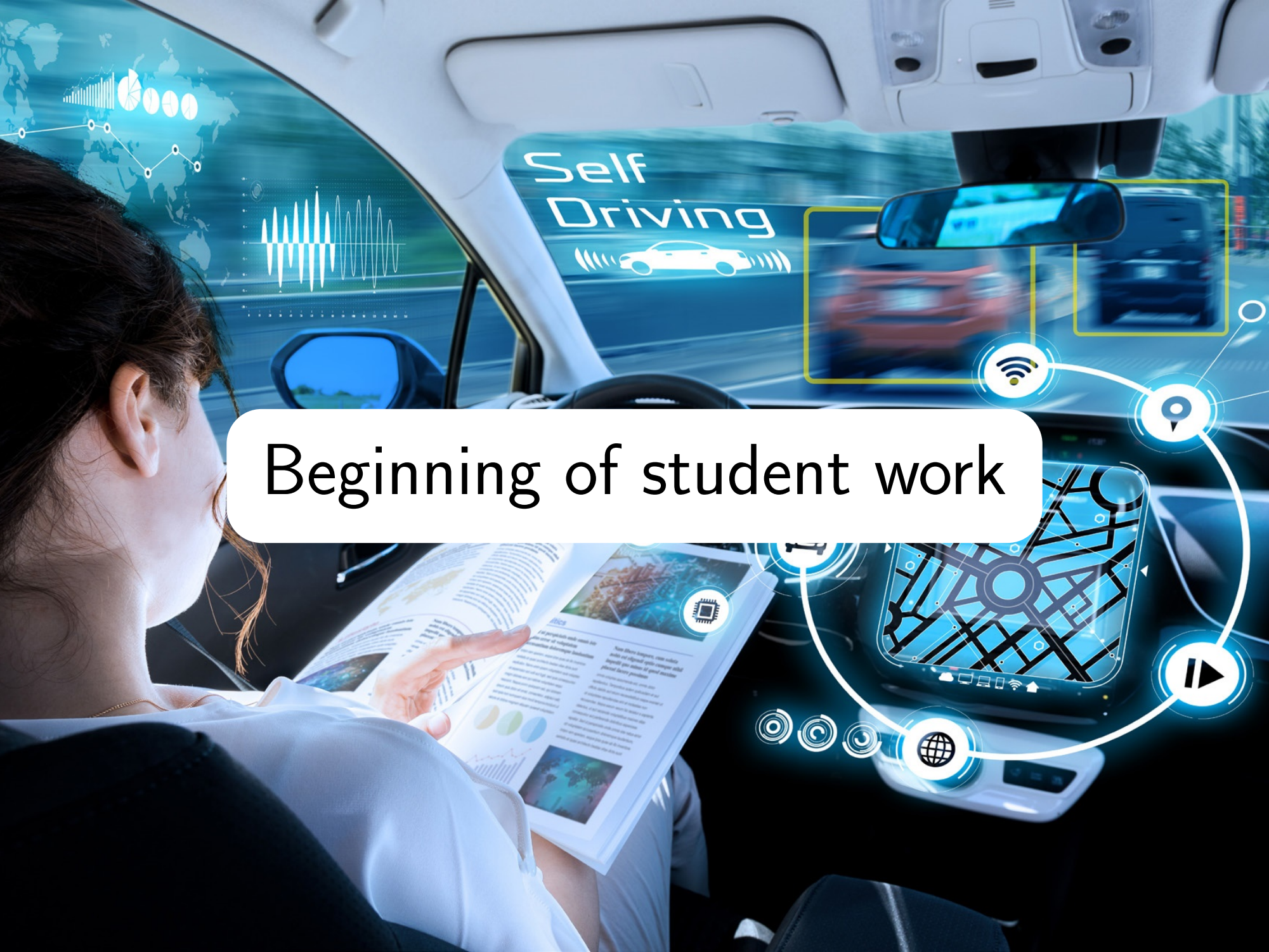
Useful functions

- for calculating calculate the position, velocity, acceleration, jerk at certain time t:

$$\text{sEval}(t, v_0, a_0, s_f, v_f, a_f, t_f) \Rightarrow s_{\text{opt}}(t)$$
$$\text{vEval}(t, v_0, a_0, s_f, v_f, a_f, t_f) \Rightarrow v_{\text{opt}}(t)$$
$$\text{aEval}(t, v_0, a_0, s_f, v_f, a_f, t_f) \Rightarrow a_{\text{opt}}(t)$$
$$\text{jEval}(t, v_0, a_0, s_f, v_f, a_f, t_f) \Rightarrow j_{\text{opt}}(t)$$

- total cost:

$$\text{totalCost}(v_0, a_0, s_f, v_f, a_f, t_f) \Rightarrow \int_0^{t_f} j_{opt}(t)^2 dt$$



Beginning of student work

Action Priming - Pass and Stop Primitives

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

Department of Industrial Engineering - University of Trento

November 22, 2022

Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Behaviours - 1/3

Intersection - actions:

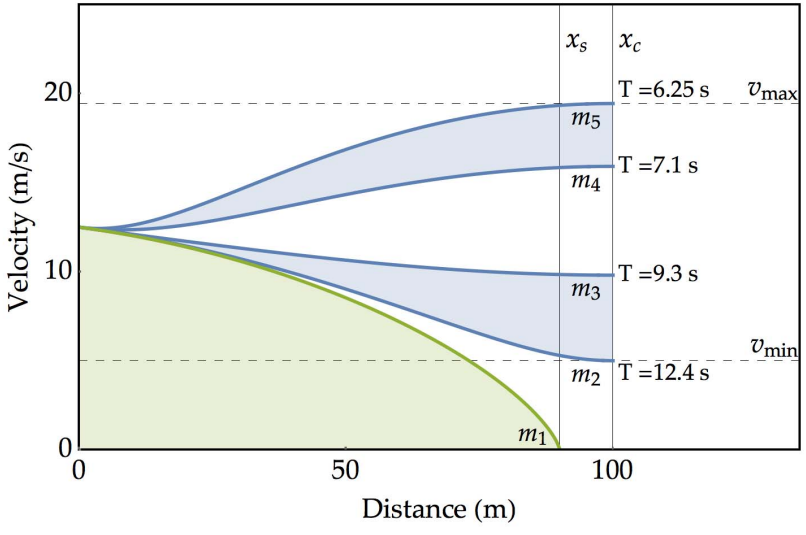
- Stopping at the stop line or earlier
- Crossing the intersection within a given speed interval and while leaving a safe time gap with the other vehicle.

Traffic-Light - actions:

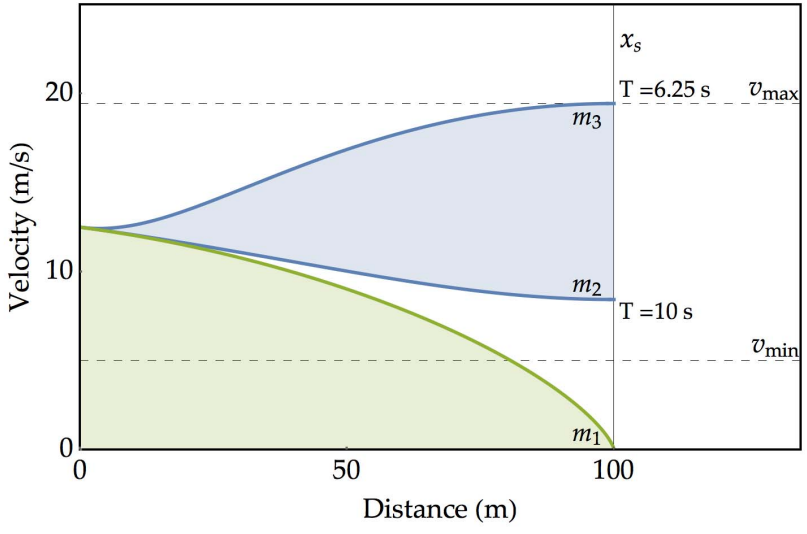
- Stopping at the traffic light or earlier
- Trespassing the traffic light within a given time interval and within a given speed interval.

Behaviours - 2/3

Intersection:

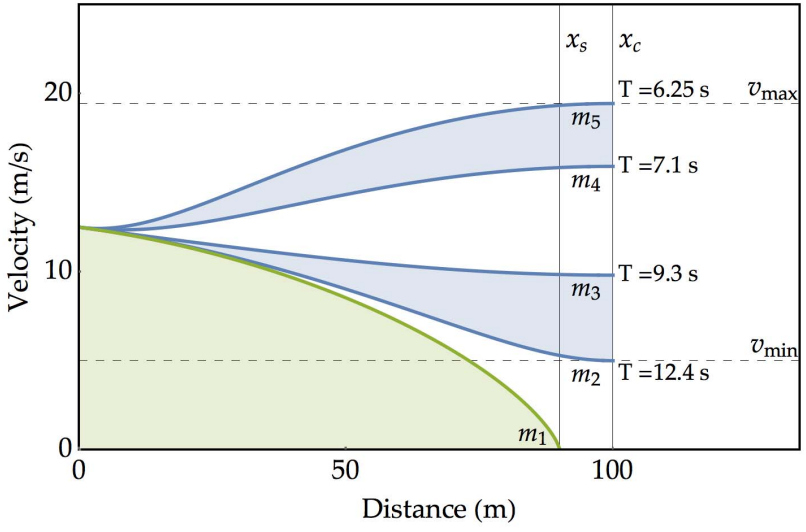


Traffic-Light:

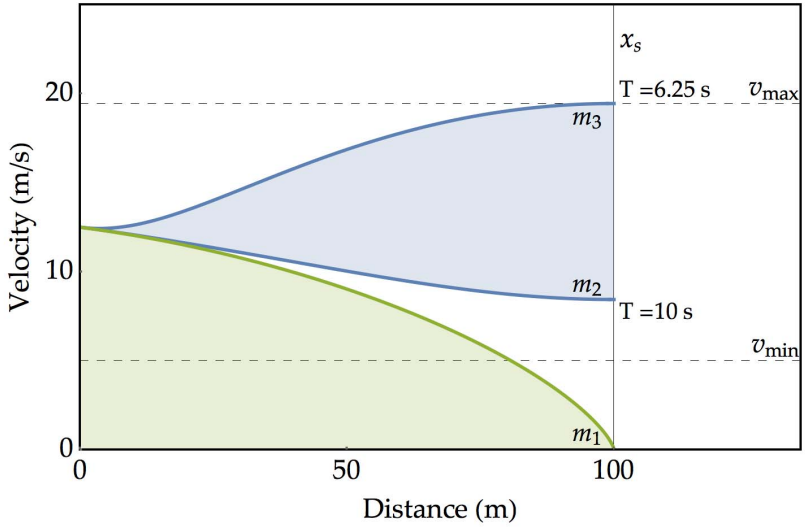


Behaviours - 3/3

Intersection:



Traffic-Light:



Motor Primitives

- Pass primitive: motor primitives trespassing the traffic light or intersection may be m_2 or m_3 (m_4,m_5), or any primitive between (the light blue shaded area).
- Stop primitives: represented either by polynomial m1 or by any primitive slower than m_1 (the green shaded area);

Advice

Despite the fact that we have the solution of the optimal control problem

There ain't no such thing as a free lunch!!

Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Stopping primitive - 1/4

evalPrimitiveCoeffs$(v_0, a_0, s_f, v_f, a_f, t_f)$

Final Conditions

- $a_f = 0$ final acceleration equal to zero
- $v_f = 0$ final velocity equal to zero
- s_f equal to the position of the traffic light or the intersection $\underline{s_f}$

Initial Conditions

- v_0 given by the sensor $\underline{v_0}$
- a_0 given by the sensor $\underline{a_0}$

We miss t_f!!

Stopping primitive - 2/4

Find the optimal time solution

The total cost is derived with respect to the final time t_f and imposed equal to zero

$$\frac{\partial \text{ totalCost}(v_0, a_0, s_f, 0, 0, t_f)}{\partial\, t_f} = 0$$

We solve the equation to find the optimal time to stop

$$\underline{T_{x_f}} = \frac{10 s_f}{2v_0 + \sqrt{4v_0^2 + 5a_0 s_f}}$$

We define a function to find the optimal time given a final position and $v_f = 0$

$$\text{finalOptTimeStop}(v_0, a_0, s_f) \Rightarrow \underline{T_{x_f}}$$

Stopping primitive - 3/4

Condition to the real solution

In order to have a real solution in the case $a_0 < 0$

$$s_f \leq -\frac{4v_0^2}{5a_0}$$

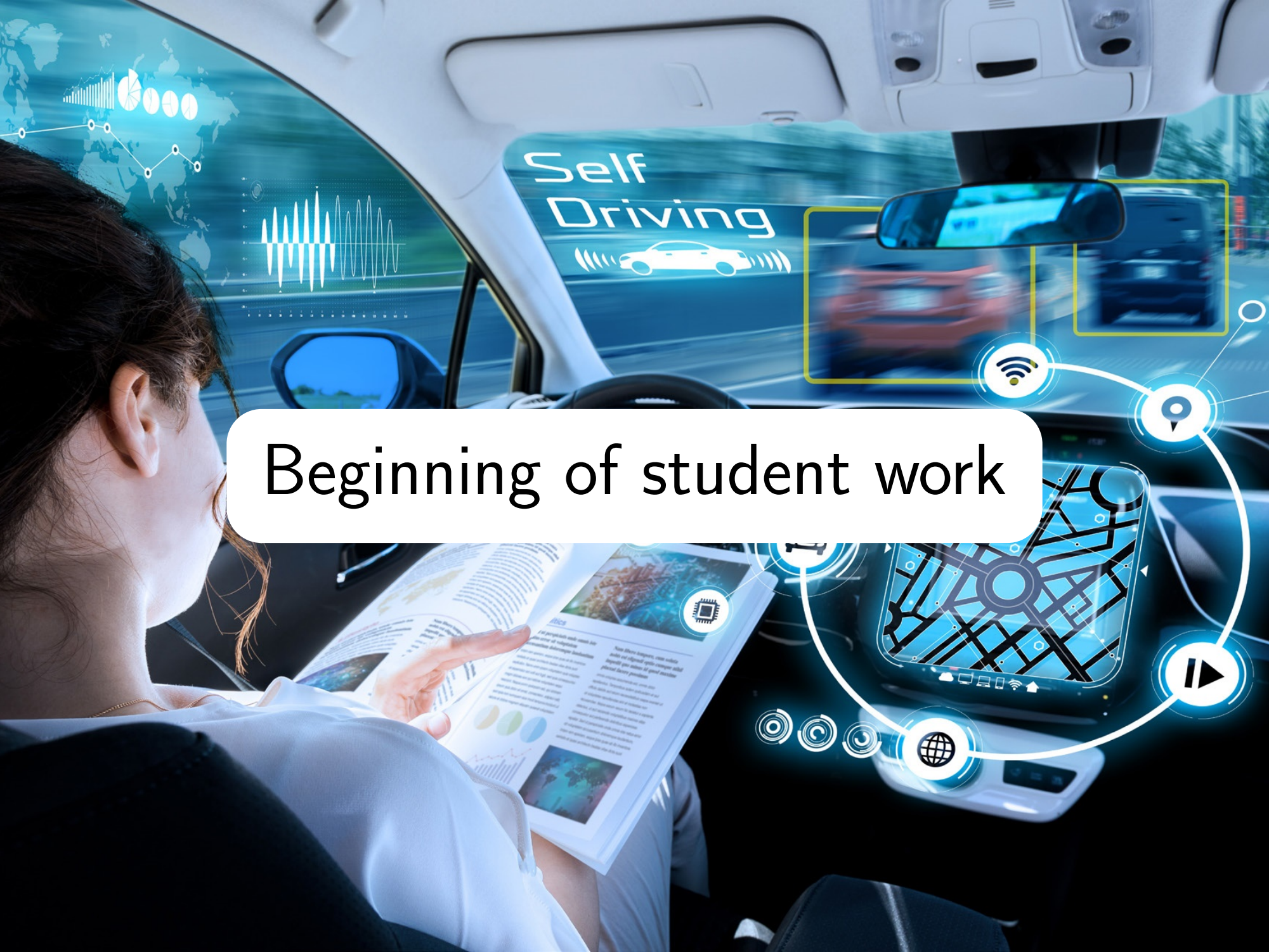
Therefore given an initial condition v_0, a_0 there is a maximum reachable final position.

Stopping primitive - 4/4

input: a_0, v_0, s_f

output: m

if $v_0 \leq 0$ or $s_f = 0$ **then**
  $m \leftarrow \varnothing$
end if
if $4v_0^2 + 5a_0 s_f < 0$ **then**
  $s_{max} = -\frac{4v_0^2}{5a0}$
  $t_f = \frac{10 s_{max}}{2v0}$
else
  $s_{max} = s_f$
  $t_f = \text{finalOptTimeStop}(v_0, a_0, s_{max})$
end if
$m \leftarrow \text{evalPrimitiveCoeffs}(v_0, a_0, s_{max}, 0, 0, t_f)$



Beginning of student work

Pass primitive - 1/8

evalPrimitiveCoeffs$(v_0, a_0, s_f, v_f, a_f, t_f)$

Final Conditions

- $a_f = 0$ final acceleration equal to zero
- v_f final velocity is within a min and a max value $v_f \in [v_{min}, v_{max}]$
- s_f equal to the position of the traffic light or the intersection
- t_f needs to be within a certain interval $t_f \in [T_{min}, T_{max}]$

Initial Conditions

- v_0 given by the sensor
- a_0 given by the sensor

Pass primitive - 2/8

In order to find the pass primitive we need two functions:

Useful Functions

- **finalOptVel** function that return the optimal final velocity given a final time
- **finalOptTime** function to find the optimal time given a final velocity

Pass primitive - 3/8

Find the optimal final velocity in terms of t_f

The total cost is derived with respect to the final velocity v_f and imposed equal to zero

$$\frac{\partial \text{ totalCost}(v_0, a_0, s_f, v_f, 0, t_f)}{\partial \, v_f} = 0$$

We solve the equation to find the optimal final velocity given a t_f

$$\underline{v_f} = \frac{15}{8}\frac{s_f}{t_f} - \frac{a_0 t_f}{8} - \frac{7v_0}{8}$$

We define a function to find the optimal velocity given the final position and the final time

$$\text{finalOptVel}(v_0, a_0, s_f, t_f) \Rightarrow \underline{v_f}$$

Pass primitive - 4/8

Find the optimal time in terms of v_f

The total cost is derived with respect to the final velocity v_f and imposed equal to zero

$$\frac{\partial \text{ totalCost}(v_0, a_0, s_f, v_f, 0, t_f)}{\partial \ v_f} = 0$$

We solve the equation to find the optimal time to reach a v_f

$$\underline{T_{v_f}} = \frac{30 s_f}{7v_0 + 8v_f + \sqrt{60 a_0 s_f + (7v_0 + 8v_f)^2}}$$

We define a function to find the optimal time given a final position and final velocity

$$\text{finalOptTime}(v_0, a_0, s_f, v_f) \Rightarrow \underline{T_{v_f}}$$

Pass primitive - 5/8

Condition to be imposed

- $v_f \in [v_{min}, v_{max}]$
- $t_f \in [T_{min}, T_{max}]$

The both the conditions can be expressed in time using

$$\text{finalOptTime}(v_0, a_0, s_f, v_f)$$

Rewritten condition using time

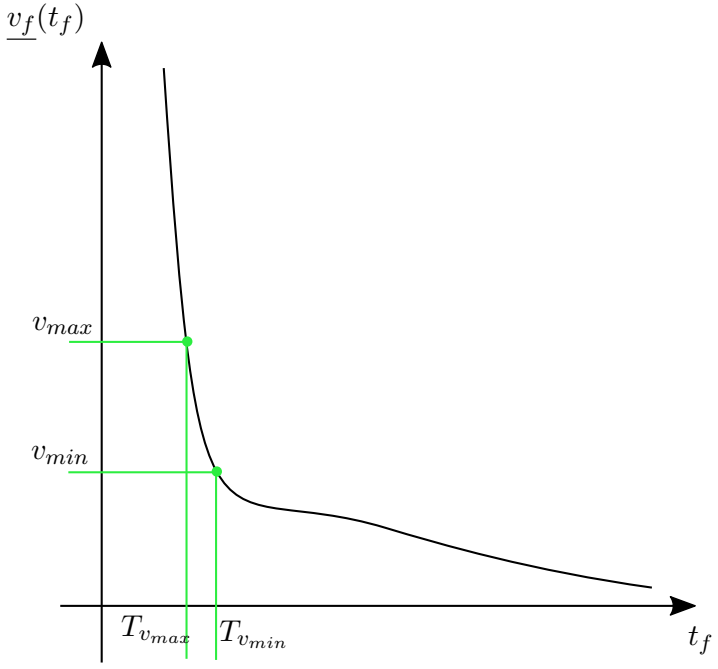
$$t_f \in [T_1, T_2] = [T_{min}, T_{max}] \cap [T_{v_{max}}, T_{v_{min}}]$$
$$T_{v_{min}} = \text{finalOptTime}(v_0, a_0, s_f, v_{min})$$
$$T_{v_{max}} = \text{finalOptTime}(v_0, a_0, s_f, v_{max})$$

Pass primitive - 6/8

Investigating the solution $\underline{v_f}$ there is two possible behaviour:

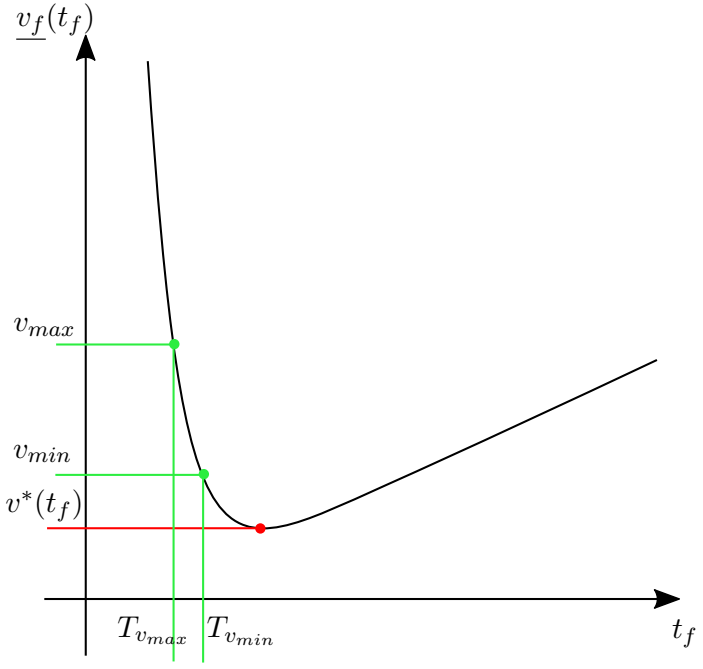
$a_0 >= 0$

There is always a solution



$a_0 < 0$

There is a minimal final velocity



Pass primitive - 7/8

Find the minimal final velocity when $a_0 < 0$

The function **finalOptVel** is derived with respect to the final time t_f and imposed equal to zero

$$\frac{\partial \text{ finalOptVel}(v_0, a_0, s_f, t_f)}{\partial\, t_f} = 0$$

We solve the equation to find the time to reach the minimum final velocity and the minimum velocity

$$T^* = \sqrt{\frac{15 s_f}{-a_0}} \quad v^* = \frac{1}{8}\left(2\sqrt{15}\sqrt{-a_0 s_f} - 7v_0\right)$$

v^* limits the minimum final velocity that we can reach. If $v^* > v_{min}$ we used v^* instead v_{min}. If $v^* > v_{max}$ there is no solution to the problem we need to stop.

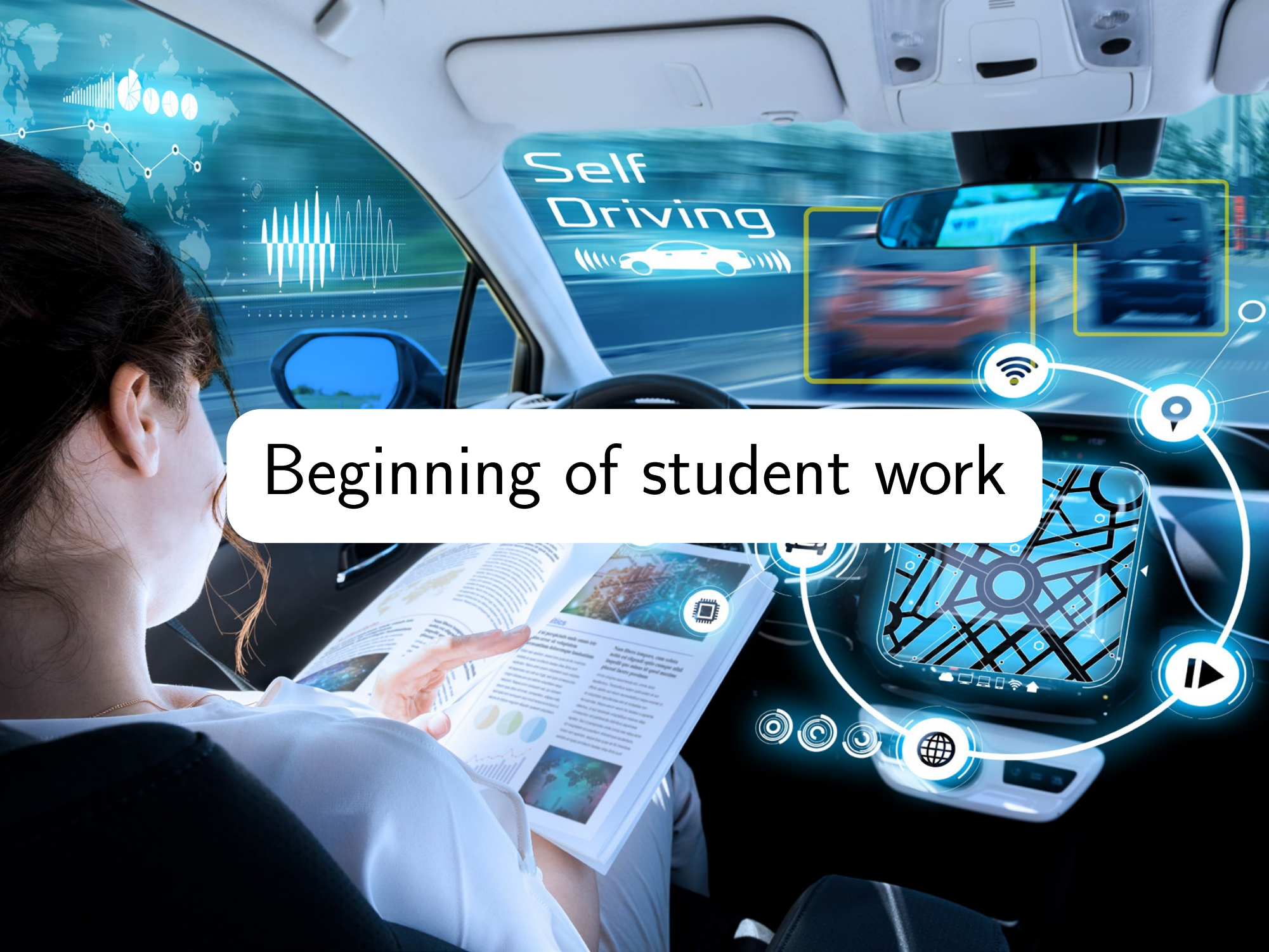
Passing primitive - 8/8 I

input: $a_0, v_0, s_f, v_{min}, v_{max}, T_{min}, T_{max}$
output: m_1, m_2

if $a_0 \geq 0$ **then**
  $T_{v_{min}} = \text{finalOptTime}(v_0, a_0, s_f, v_{min})$
  $T_{v_{max}} = \text{finalOptTime}(v_0, a_0, s_f, v_{max})$
else
  $T^* = \sqrt{\frac{15 s_f}{-a_0}}$
  $v^* = \frac{1}{4}\left(\sqrt{15 - a_0 s_f} - 7 v_0\right)$
  if $v^* < v_{min} < v_{max}$ **then**
    $T_{v_{min}} = \text{finalOptTime}(v_0, a_0, s_f, v_{min})$
    $T_{v_{max}} = \text{finalOptTime}(v_0, a_0, s_f, v_{max})$
  else if $v_{min} < v^* < v_{max}$ **then**
    $T_{v_{min}} = T^*$
    $T_{v_{max}} = \text{finalOptTime}(v_0, a_0, s_f, v_{max})$
  else

Passing primitive - 8/8 II

$T_{v_{min}} = 0$
$T_{v_{max}} = 0$
end if
end if
$[T_1, T_2] = [T_{min}, T_{max}] \cap [T_{v_{max}}, T_{v_{min}}]$
if $0 < T_1 <= T_2$ **then**
$v_{min} = \text{finalOptVel}(v_0, a_0, s_f, T_2)$
$v_{max} = \text{finalOptVel}(v_0, a_0, s_f, T_1)$
$m_1 \leftarrow \text{evalPrimitiveCoeffs}(v_0, a_0, s_f, v_{max}, 0, T_1)$
$m_2 \leftarrow \text{evalPrimitiveCoeffs}(v_0, a_0, s_f, v_{min}, 0, T_2)$
else
$m_1 \leftarrow \varnothing$
$m_2 \leftarrow \varnothing$
end if



Beginning of student work

Action Selection

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

Department of Industrial Engineering - University of Trento

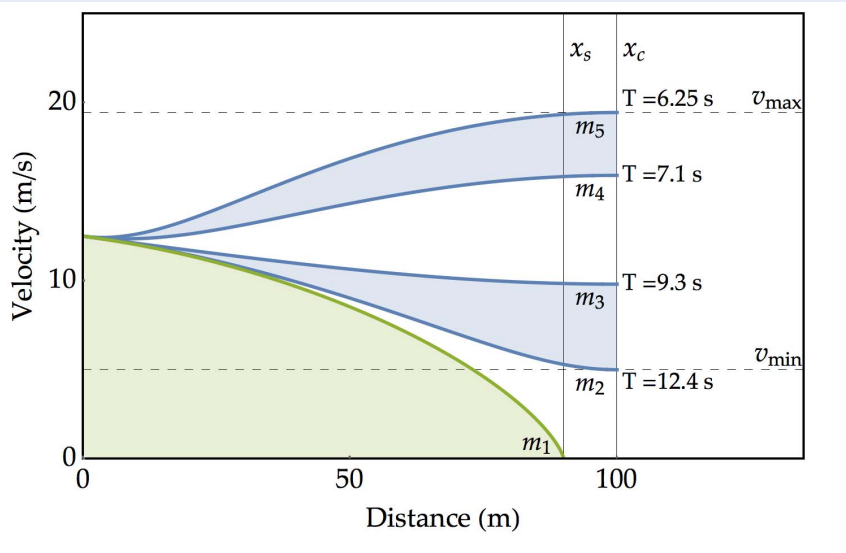
November 22, 2022

Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Behaviours Recap

Intersection

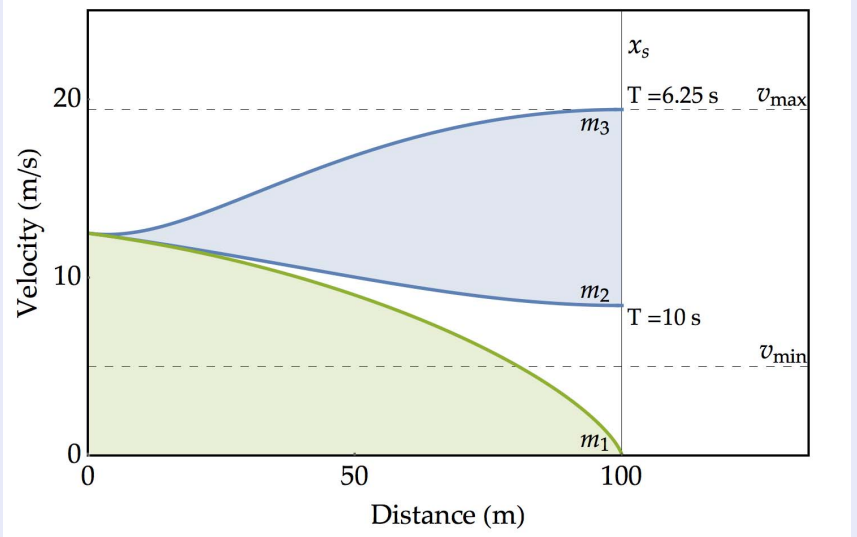


Passing$(a_0, v_0, s_f, T_{min_a}, T_{max_a}, v_{min}, v_{max})$

Passing$(a_0, v_0, s_f, T_{min_b}, T_{max_b}, v_{min}, v_{max})$

Stopping(a_0, v_0, s_f)

Traffic-Light



Passing$(a_0, v_0, s_f, v_{min}, v_{max}, T_{min}, T_{max})$

Stopping(a_0, v_0, s_f)

The behaviours output a motor primitive $m = \{c_1, c_2, c_3, c_4, c_5\}$

Action representation - 1/2

The Coefficients of the Motor Primitive

The state of the system

$$c_1 \text{ is } v_0$$

$$c_2 \text{ is } a_0$$

The control

$$j(t) = c_3 + c_4 t + \frac{1}{2} c_5 t^2, \quad t \in [0, t_f]$$

c_3 is the jerk in particular $j(0)$ instantaneous control

c_4 is the snap

c_5 is the crackle

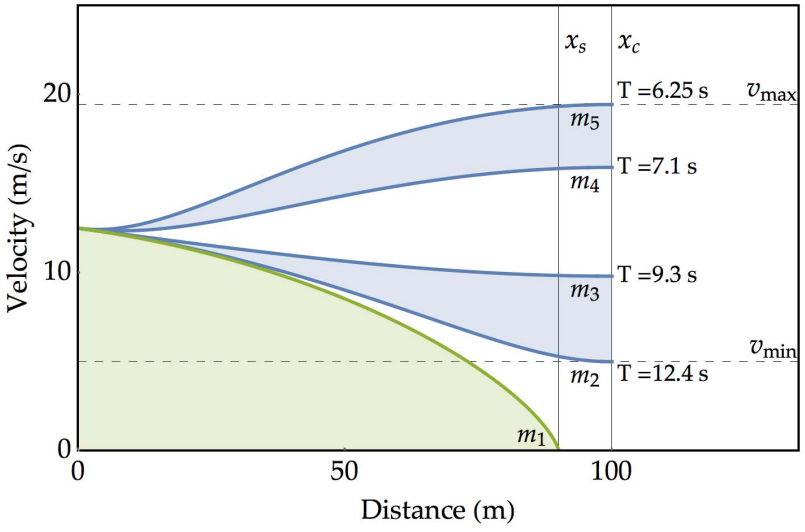
Action representation - 2/2

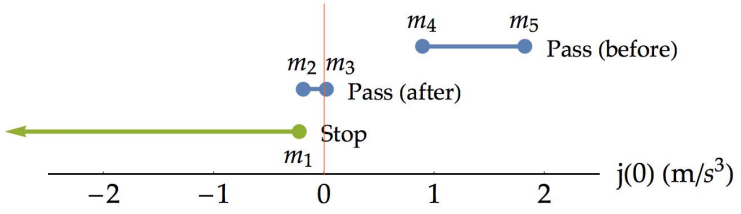
Receding-horizon framework

Motor plans are continuously updated to adapt to the changing environment.
$j(0)$ is the instantaneous control.

Representation of the control

$j(0)$ is the instantaneous control distinct motor primitives can share the same $j(0)$





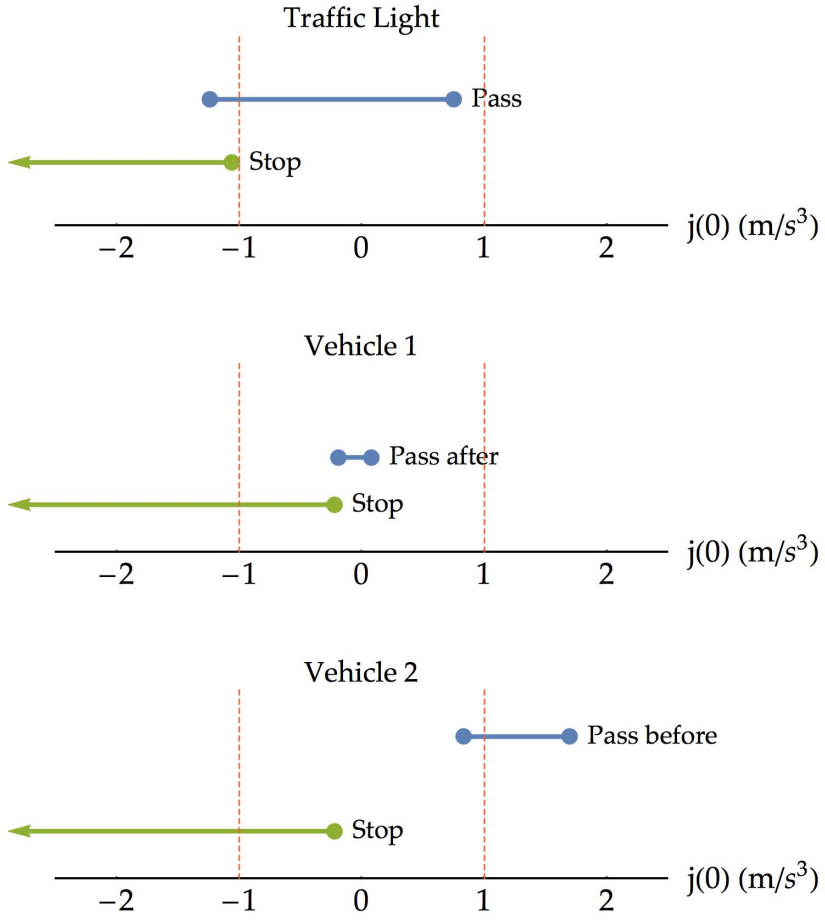
Inhibition mechanism

Inhibition mechanism

Assume that one of the actions lumped in $j(0)$ is hazardous.

We suppose that a dangerous action inhibits all other actions that share the same $j(0)$;

A hazardous action blocks all actions that would be produced with the same instantaneous control.

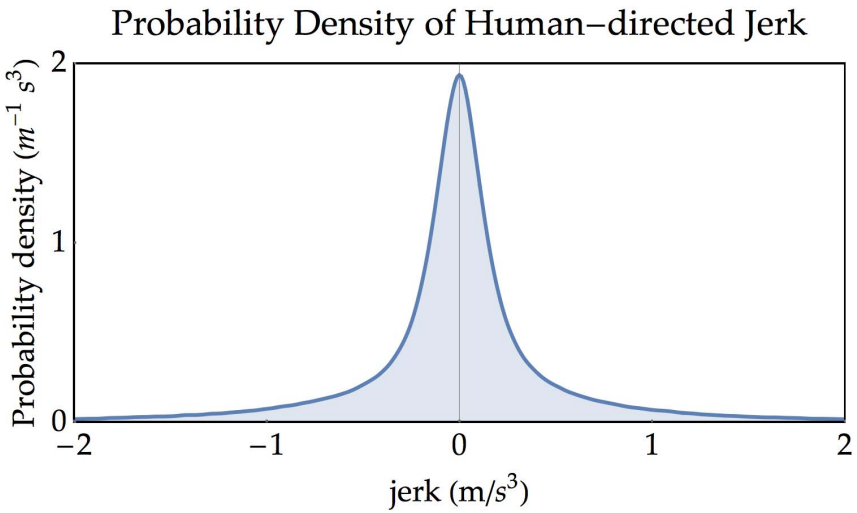


Experimental Human-Directed Jerk[1]

Human-Directed Jerk

The interval $\left[-1\frac{m}{s^3}, 1\frac{m}{s^3}\right]$ accounts for slightly more than 90% of observations

99.3% of observations lie in the interval $\left[-3\frac{m}{s^3}, 3\frac{m}{s^3}\right]$.



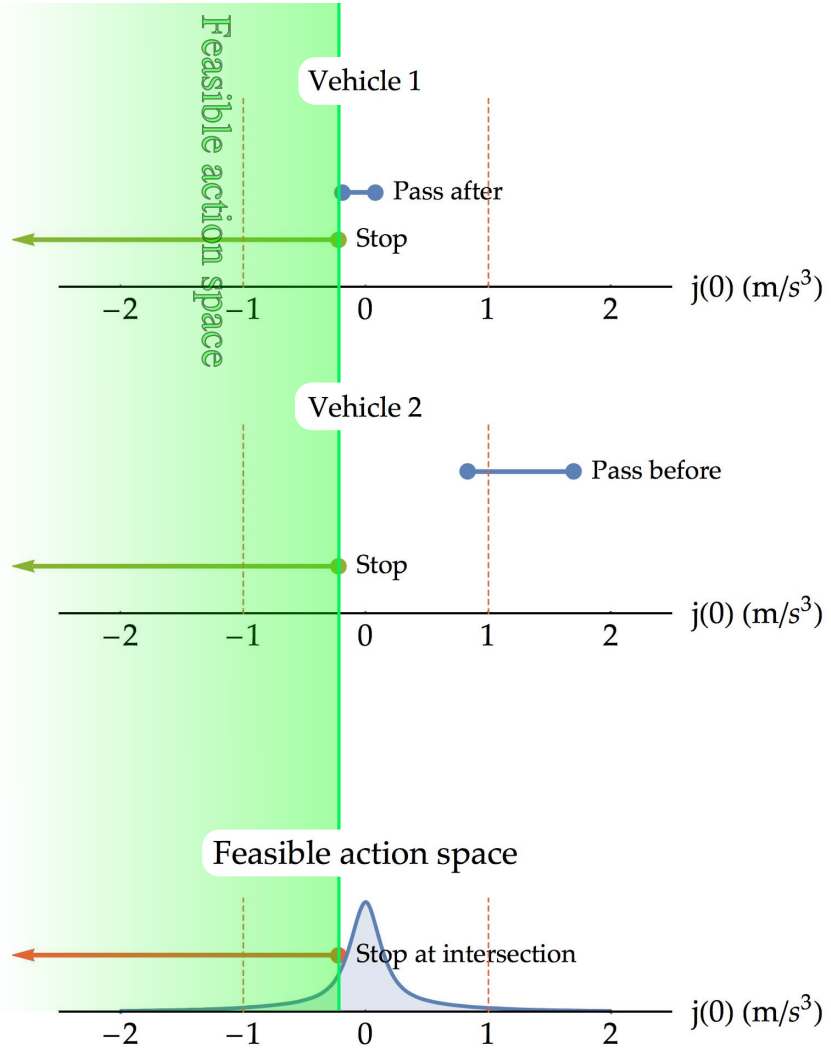
[1] Mauro Da Lio et al. "Biologically guided driver modeling: The stop behavior of human car drivers". In: *IEEE Transactions on Intelligent Transportation Systems* 19.8 (2018).

Mirroring

Human-Directed Jerk

We use the minimal intervention principle.

We choose the smaller absolute jerk as j_0



Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

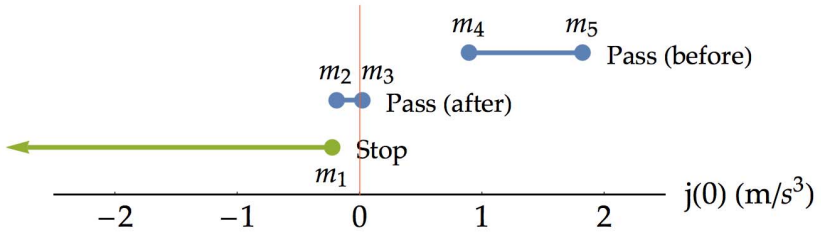
Generation of the motor primitive - 1/3

Motor primitive for j_0 range definition

These motor primitives define the admissible range for j_0

Passing$(a_0, v_0, s_f, v_{min}, v_{max}, T_{min}, T_{max})$

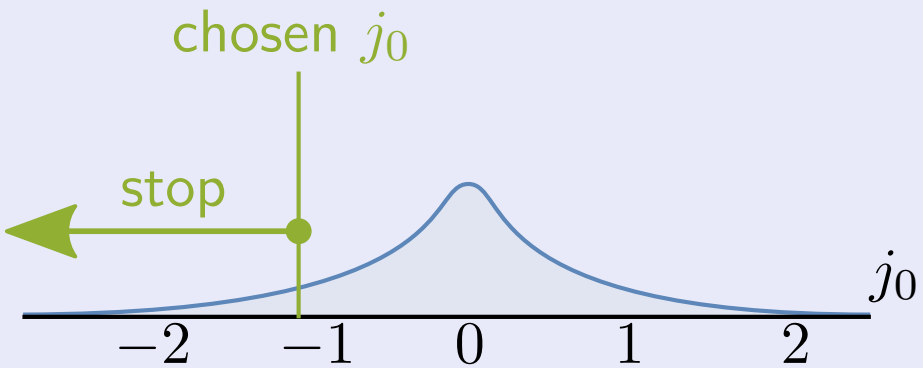
Stopping(a_0, v_0, s_f)



We identify four cases.

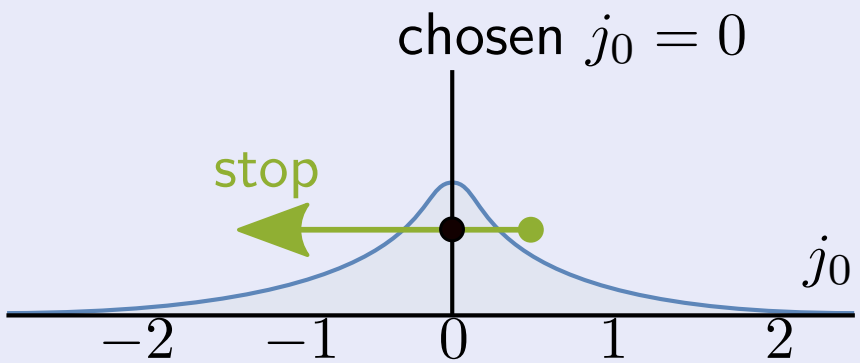
Generation of the motor primitive - 2/3

Case 1: Stop primitive



We use the Stop primitive already calculated

Case 2: Stop primitive

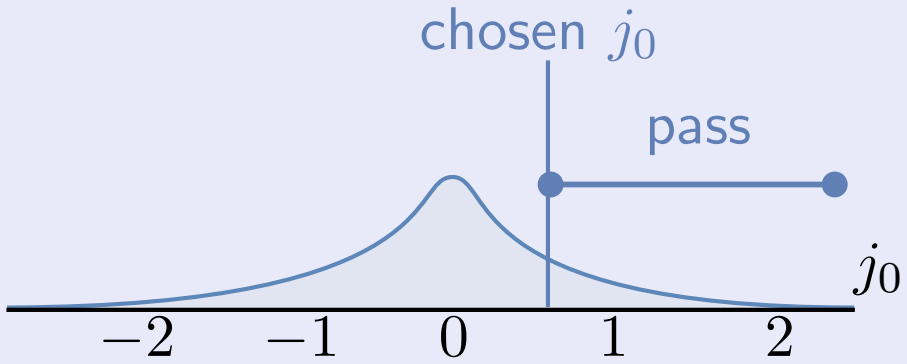


We need to calculate the Stop primitive with $j_0 = 0$

To obtain this primitive we need to determine a Stop primitive function of

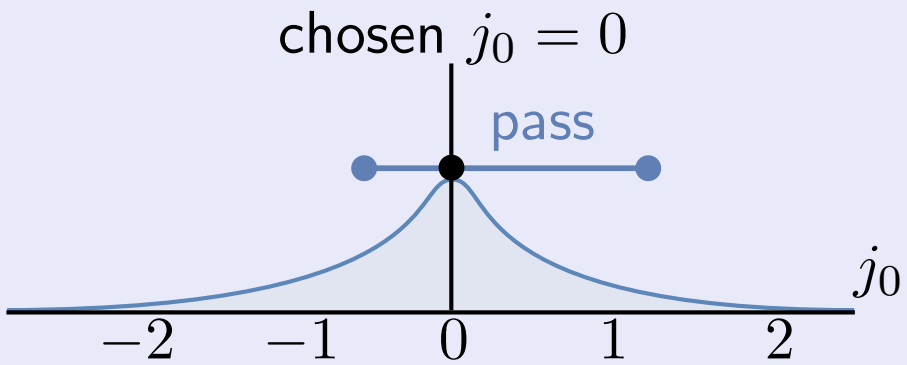
Generation of the motor primitive - 3/3

Case 3: Pass primitive



We use the Pass primitive already calculated

Case 4: j_0 within the pass primitive range



We need to calculate the Pass primitive with $j_0 = 0$

To obtain this primitive we need to determine a Pass primitive function of

Generation of Stop primitive $j_0 = 0$ - 1/2

Find the position velocity function of the initial jerk $j_0 = 0$

The optimal control is imposed equal to 0

$$\text{jEval}(t, v_0, a_0, s_f, 0, 0, t_f) = j_0 = 0$$

The solution is simplified thanks to the hypothesis of $j_0 = 0$. We solve the equation to find the final position

$$\underline{s_f^{j_0}} = \frac{1}{60} t_f \left(9 a_0 t_f + 36 v_0\right)$$

We define a function to find the final position with $j_0 = 0$

$$\text{finalOptPosj0}(v_0, a_0, t_f) \Rightarrow \underline{s_f^{j_0}}$$

Generation of Stop primitive $j_0 = 0$ - 2/2

Find the optimal time

The function **finalOptTimeStop** with $s_f = \underline{s_f^{j_0}}$ is imposed equal to t_f

$$\text{finalOptTimeStop}(v_0, a_0, \text{finalOptPosj0}(v_0, a_0, t_f)) = t_f$$

We solve the equation to find the optimal time to reach a $\underline{s_f^{j_0}}$

$$\underline{T_{s_f^{j_0}}} = -\frac{2v_0}{a0}$$

We define a function to find the optimal time given a final position that impose $j_0 = 0$

$$\text{finalOptTimeStopj0}(v_0, a_0) \Rightarrow \underline{T_{s_f^{j_0}}}$$

Stop primitive with $j_0 = 0$

input: a_0, v_0, s_f
output: m

1: **if** $v_0 > 0$ and $a < 0$ **then**
2: $T = \text{finalOptTimeStopj0}(v_0, a_0)$
3: $s_{max} = \text{finalOptPosj0}(v_0, a_0, T)$
4: $m \leftarrow \text{evalPrimitiveCoeffs}(v_0, a_0, s_{max}, 0, 0, T)$
5: **else**
6: $T = 0$
7: $s_{max} = 0$
8: $m \leftarrow \varnothing$
9: **end if**

Generation of Pass primitive $j_0 = 0$ - 1/2

Find the final velocity function of the initial jerk $j_0 = 0$

The optimal control is imposed equal to 0

$$\text{jEval}(t, v_0, a_0, s_f, v_f, 0, t_f) = j_0 = 0$$

The solution is simplified thanks to the hypothesis of $j_0 = 0$. We solve the equation to find the optimal final velocity as function of j_0

$$\underline{v_f^{j_0}} = \frac{1}{8}\left(\frac{20 s_f}{t_f} - t_f 3 a_0 - 12 v_0\right)$$

We define a function to find the final velocity with $j_0 = 0$

$$\text{finalOptVelj0}(v_0, a_0, s_f, t_f) \Rightarrow \underline{v_f^{j_0}}$$

Generation of Pass primitive $j_0 = 0$ - 2/2

Find the optimal time in terms of $v_f^{j_0}$

The function **finalOptTime** is imposed equal to f_f

$$\text{finalOptTime}(v_0, a_0, s_f, \text{finalOptVelj0}(v_0, a_0, s_f, t_f)) = t_f$$

We solve the equation to find the optimal time to reach a $v_f^{j_0}$

$$\underline{T_{v_f^{j_0}}} = \left\{ \frac{10 s_f}{5v_0 - \sqrt{5}\sqrt{8a_0 s_f + 5v_0^2}}, \frac{10 s_f}{\sqrt{5}\sqrt{8a_0 s_f + 5v_0^2} + 5v_0} \right\}$$

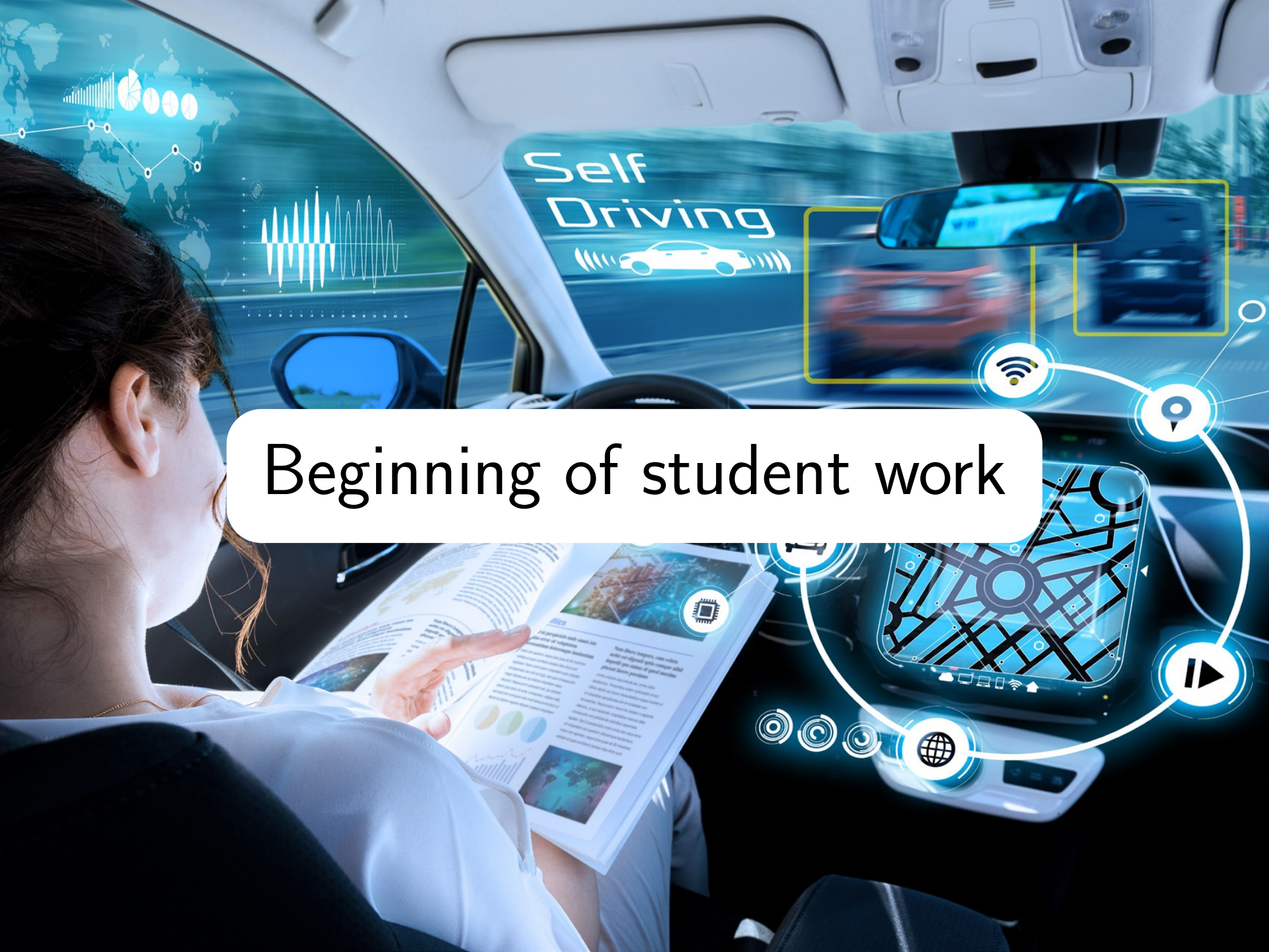
We define a function to find the optimal time given a final position that impose $j_0 = 0$

$$\text{finalOptTimej0}(v_0, a_0, s_f) \Rightarrow \underline{T_{v_f^{j_0}}}$$

Passing primitive with $j_0 = 0$

input: $a_0, v_0, s_f, v_{min}, v_{max}$
output: m_1

1: $T_1 = \text{finalOptTimej0}(v_0, a_0, s_f)\,[1]$
2: $v_1 = \text{finalOptVelj0}(v_0, a_0, s_f, T_1)$
3: **if** $v_{min} < v_1 < v_{max}$ **then**
4: $\quad m_1 \leftarrow \text{evalPrimitiveCoeffs}(v_0, a_0, s_f, v_1, 0, T_1)$
5: **else**
6: $\quad T_2 = \text{finalOptTimej0}(v_0, a_0, s_f)\,[2]$
7: $\quad v_2 = \text{finalOptVelj0}(v_0, a_0, s_f, T_2)$
8: $\quad$ **if** $v_{min} < v_2 < v_{max}$ **then**
9: $\quad\quad m_1 \leftarrow \text{evalPrimitiveCoeffs}(v_0, a_0, s_f, v_2, 0, T_2)$
10: $\quad$ **else**
11: $\quad\quad m_1 \leftarrow \varnothing$
12: $\quad$ **end if**
13: **end if**



Beginning of student work

Low level control

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

Department of Industrial Engineering - University of Trento

November 22, 2022

Lessons - outline

Low Level Control

From the primitive to the control signal

- from the primitive we have the best primitive as $m^* = \{c_1, c_2, c_3, c_4, c_5\}$
- we needs a function that generate the requested acceleration a_{req}.

Low Level Control

Trivial but wrong

$$a_{req} = \text{aEval}(\Delta t, m^*)$$

If the vehicle for some reasons not generates the exactly the same acceleration this low level control does not work.
This controller is not robust!

Low Level Control

Integrated jerk - forward/backward Euler

$$a_{req} = a_0 + \text{jEval}([0, \Delta t], m^*)\Delta t$$

This solution is affected from measured (estimated) acceleration (a_0)

Low Level Control

Integrated jerk - forward/backward Euler - with internal a_0

Initialization

$$\underline{a_0} = a_0$$

Integration Step

$$a_{req} = \underline{a_0} + \text{jEval}([0, \Delta t], m^*)\Delta t$$

Update internal $\underline{a_0}$ variable

$$\underline{a_0} = a_{req}$$

Low Level Control

Integrated jerk - trapezoidal - with internal a_0

Initialization

$$\underline{a_0} = a_0$$

Integration Step

$$a_{req} = \underline{a_0} + \frac{\Delta t}{2}(\text{jEval}(0, m^*) + \text{jEval}(\Delta t, m^*))$$

Update internal $\underline{a_0}$ variable

$$\underline{a_0} = a_{req}$$

Include saturation

$$a_{req} = \max(\min(a_{req}, -a_{min}), a_{max})$$

PID control

PID control

Define the error

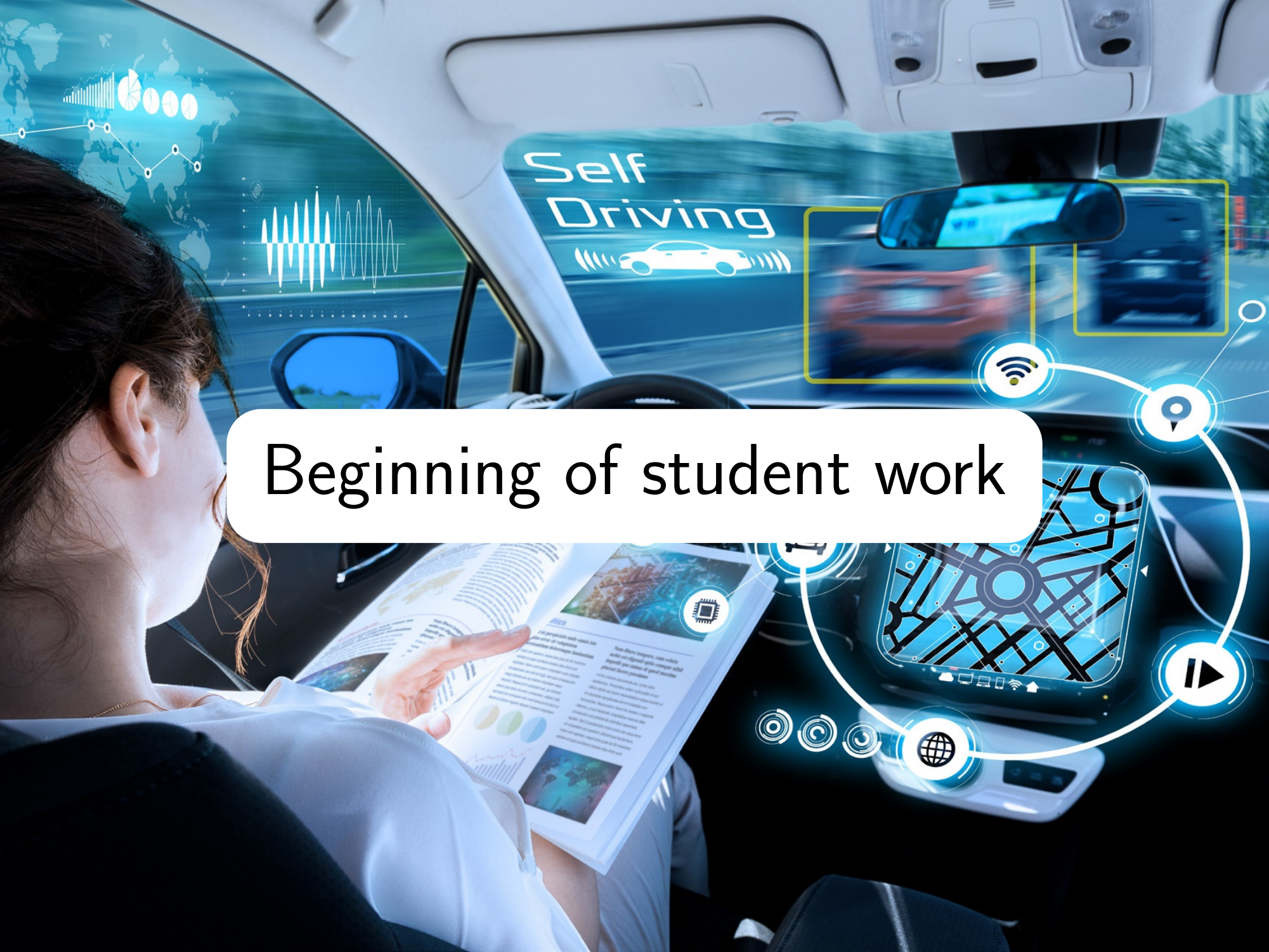
$$e = a_{req} - a$$

Requested pedal

$$R_{eq,pedal} = e * P_{gain} + e_{int} * I_{gain}$$

Compute the integral of the error

$$e_{int} = e_{int} + e * d_t$$



Beginning of student work

Main Loop

Gastone Pietro Rosati Papini
gastone.rosatipapini@unitn.it

Department of Industrial Engineering - University of Trento

November 22, 2022

Lessons - outline

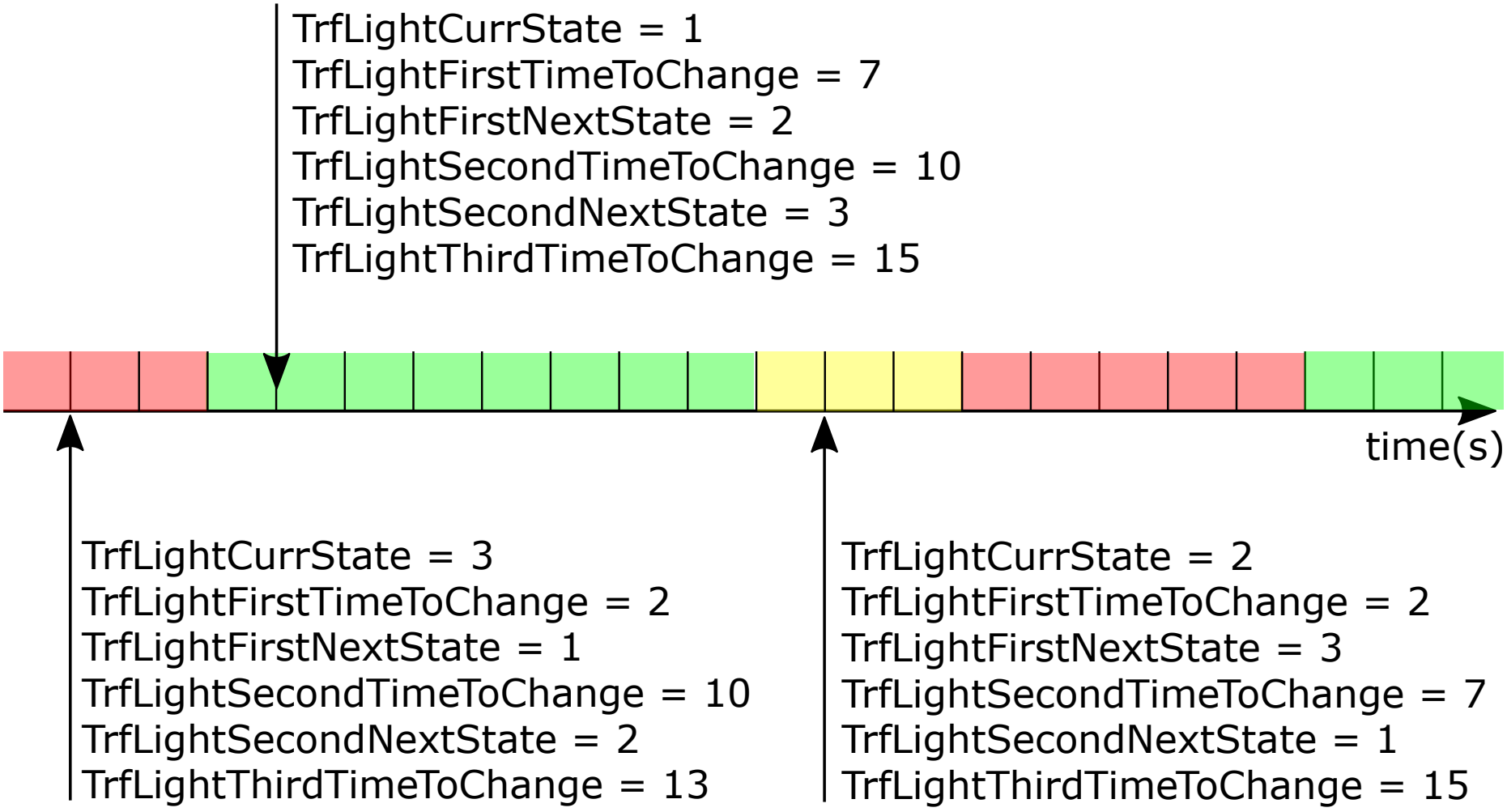
Vehicle data

- RequestedCruisingSpeed: the requested cruising speed of the vehicle
- VLgtFild: the longitudinal velocity of the vehicle
- ALgtFild: the longitudinal acceleration of the vehicle

Traffic light signals - 1/2

- NrTrfLights: the number of traffic lights in the scenario
- TrfLightDist: the distance of the traffic light from the vehicle
- TrfLightCurrState: the current state of the traffic light
- TrfLightFirstNextState: the first next state of the traffic light
- TrfLightSecondNextState: the second next state of the traffic light
- TrfLightFirstTimeToChange: the time to change to the first next state of the traffic light
- TrfLightSecondTimeToChange: the time to change to the second next state of the traffic light
- TrfLightThirdTimeToChange: the time to change to the third next state of the traffic light

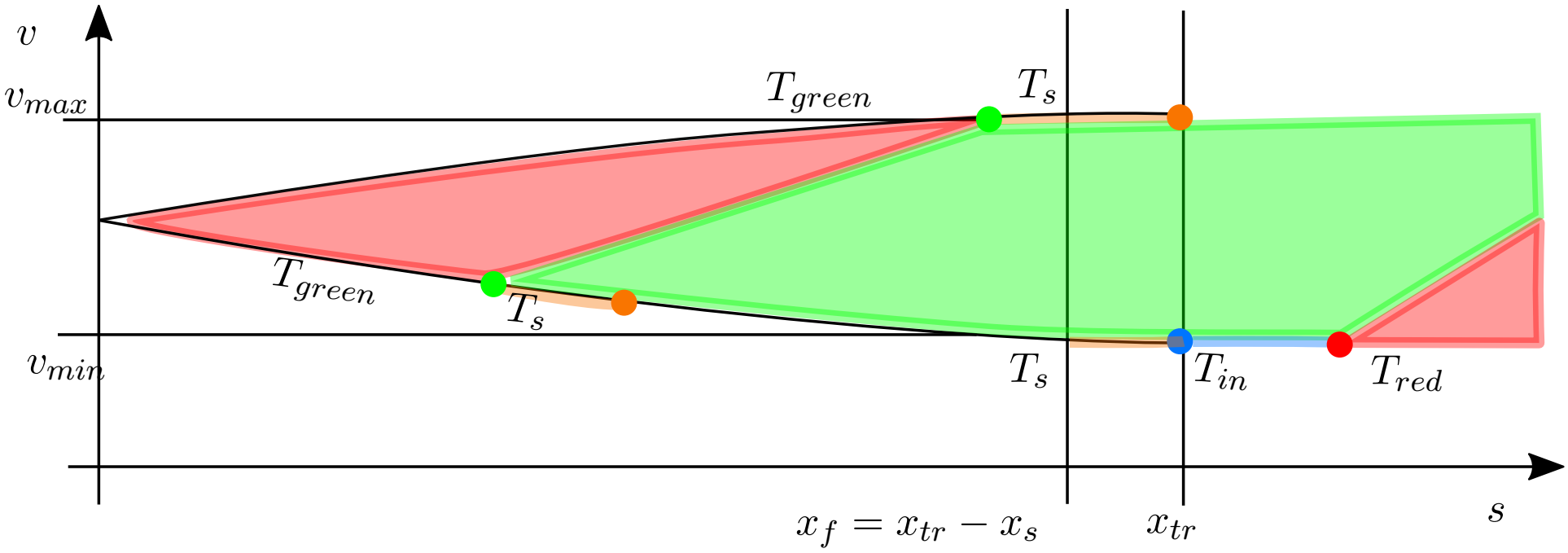
Traffic light signals - 2/2



Lessons - outline

1. Agent Definition
   - Minimum Jeck - Optimal control
   - Action Priming - Behaviours
   - Action Priming - Motor Primitives
     - Stopping primitive
     - Pass primitive
   - Action Selection
     - Action representation - Action Maps
     - Inhibition mechanism
     - Mirroring
   - Action Generation
2. Low level control
3. Interfaces
4. From the scenarios to the times

Pass primitive safety margin - 1/3



Pass primitive safety margin - 2/3

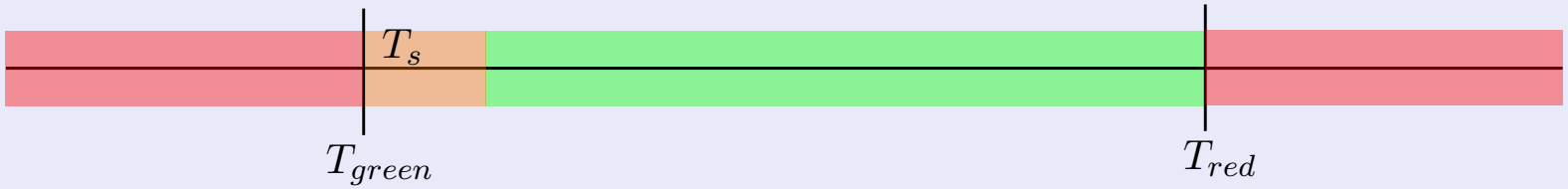
Safety space

The safety space x_s identifies the minimum distance at which the traffic light must be green if the vehicle is to pass.

Safety times

What is the maximum time to travel the safety space?

$$T_s = \frac{x_s}{v_{min}}$$



Pass primitive safety margin - 3/3

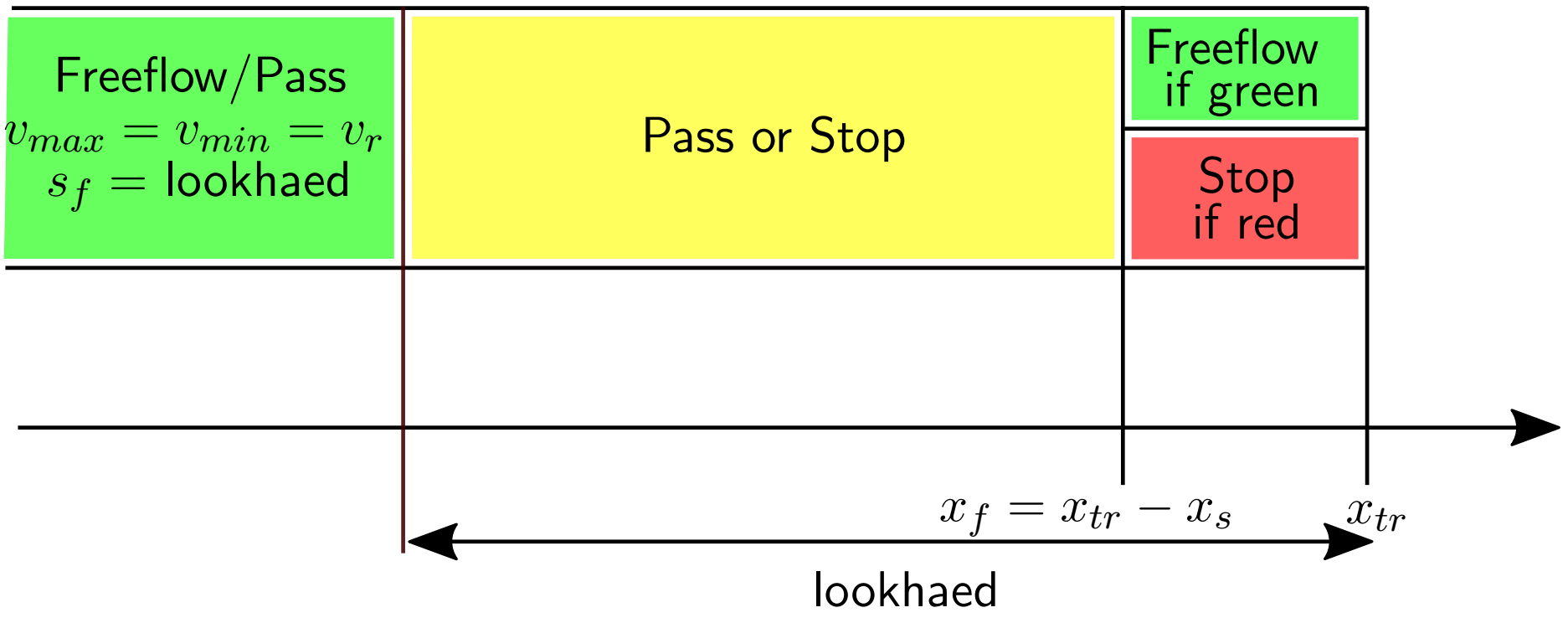
Free the intersection within the green light

Considering the lower velocity v_{min} and the x_{in} we get a time T_{in}

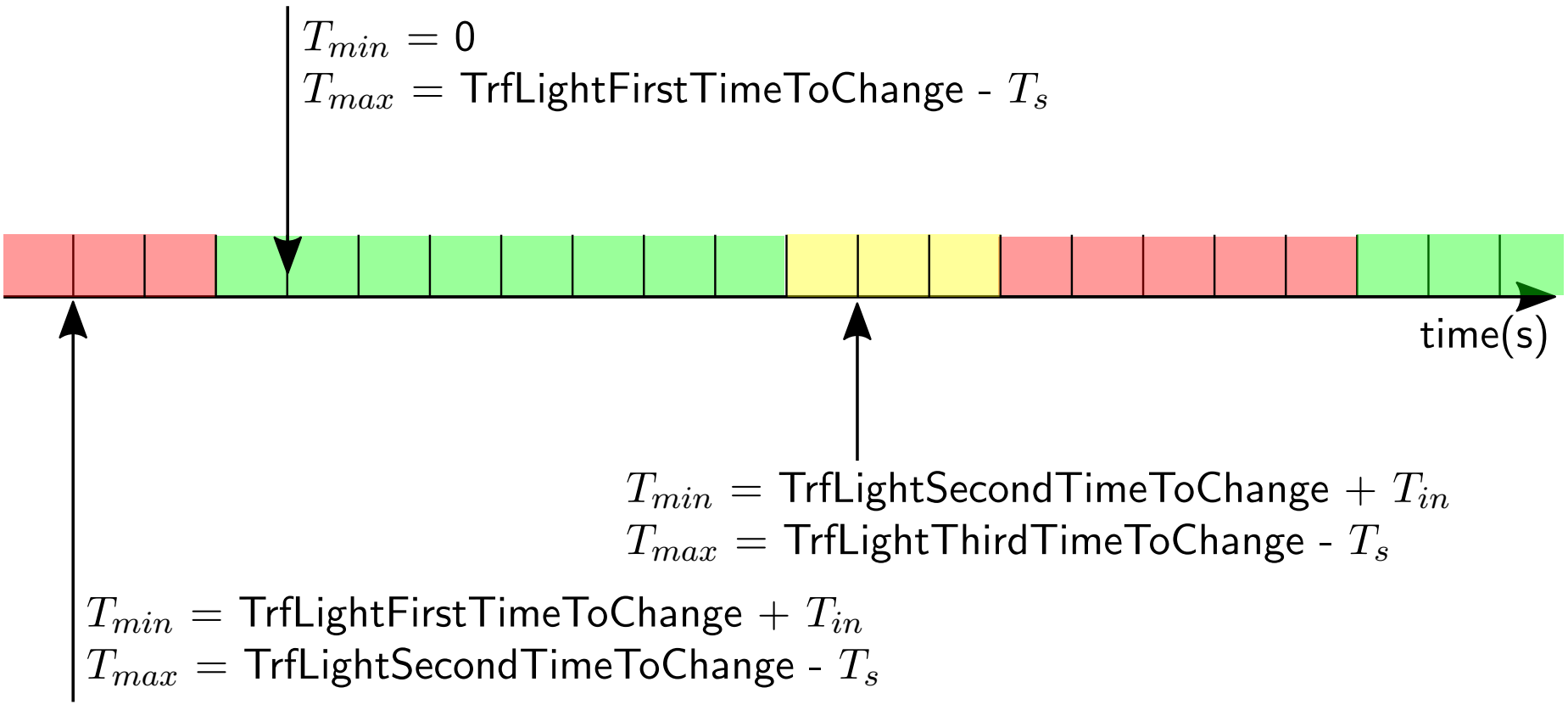
$$T_{in} = \frac{x_{in}}{v_{min}}$$

where x_{in} is the length of the intersection

High-level logic



Traffic-Light - Definition of the times



Main Logic for the traffic-light I

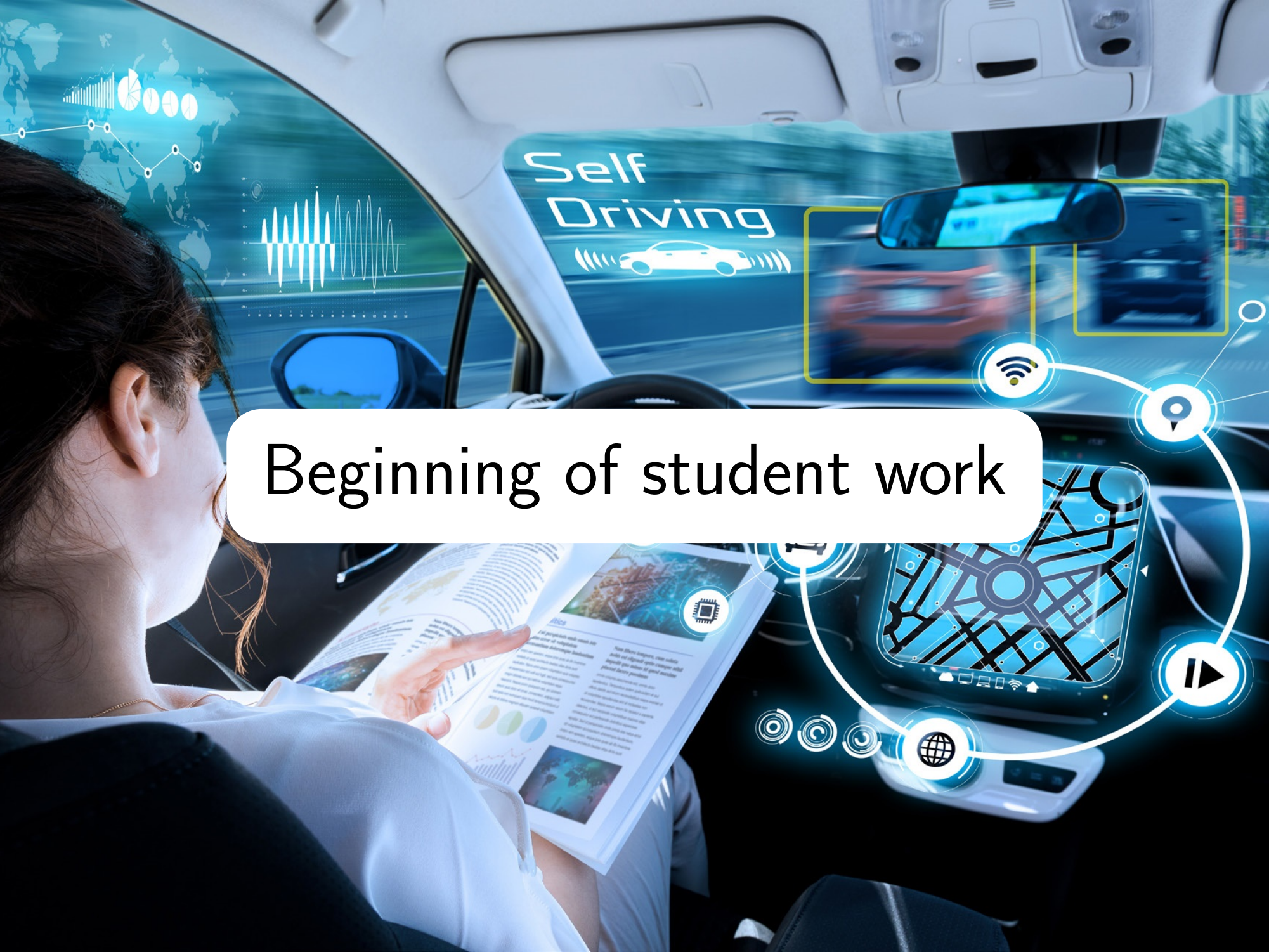
1: $v_0 =$ VLgtFild
2: $a_0 =$ ALgtFild
3: **define** FreeFlow(v_0,a_0,x_f,v_r) $\rightarrow$ Pass(v_0,a_0,x_f,v_r,v_r,0,0)
4: lookhead = Max(50,$v_0 * 5$)
5: $v_{min} = 3$
6: $v_{max} = 15$
7: $v_r =$ RequestedCruisingSpeed
8: $x_s \approx 5$
9: $T_s = x_s / v_{min}$
10: $x_{in} \approx 10$
11: $T_{in} = x_{in} / v_{min}$
12: **if** NrTrfLights != 0 **then**
13: $x_{tr} =$ TrfLightDist;
14: $x_{stop} =$ TrfLightDist - $x_s/2$;
15: **end if**
16: **if** NrTrfLights = 0 **or** $x_{tr} \geq$ lookhead **then**

Main Logic for the traffic-light II

17: $m^* =$ FreeFlow(v_0,a_0,lookhead,v_r)
18: **else**
19: **switch** (TrfLightCurrState)
20: **case** 1**:**
21: $T_{green} = 0$
22: $T_{red} =$ TrfLightFirstTimeToChange - T_{in}
23: **case** 2**:**
24: $T_{green} =$ TrfLightSecondTimeToChange $+ T_s$
25: $T_{red} =$ TrfLightThirdTimeToChange - T_{in}
26: **case** 3**:**
27: $T_{green} =$ TrfLightFirstTimeToChange $+ T_s$
28: $T_{red} =$ TrfLightSecondTimeToChange - T_{in}
29: **end switch**
30: **if** TrfLightCurrState $== 1$ **and** TrfLightDist $\leq x_s$ **then**
31: $m^* =$ FreeFlow(v_0,a_0,lookhead,v_r)
32: **else**

Main Logic for the traffic-light III

33: $\{m_1, m_2\} = \text{Pass}(v_0, a_0, x_{tr}, v_{min}, v_{max}, T_{green}, T_{red})$
34: **if** $m_1 == \varnothing$ **and** $m_2 == \varnothing$ **then**
35: $m^* = \text{Stop}(v_0, a_0, x_{stop})$
36: **else**
37: **if** $(m_1[3] < 0$ **and** $m_2[3] > 0)$ **or** $(m_1[3] > 0$ **and** $m_2[3] < 0)$ **then**
38: $m^* = \text{PassJ0}(v_0, a_0, x_{tr}, v_{min}, v_{max})$
39: **else**
40: **if** $\|m_1[3]\| < \|m_2[3]\|$ **then**
41: $m^* = m_1$
42: **else**
43: $m^* = m_2$
44: **end if**
45: **end if**
46: **end if**
47: **end if**
48: **end if**



Beginning of student work