Tutorials (*Travaux dirigés*), **mono mobile-robot sub-part**

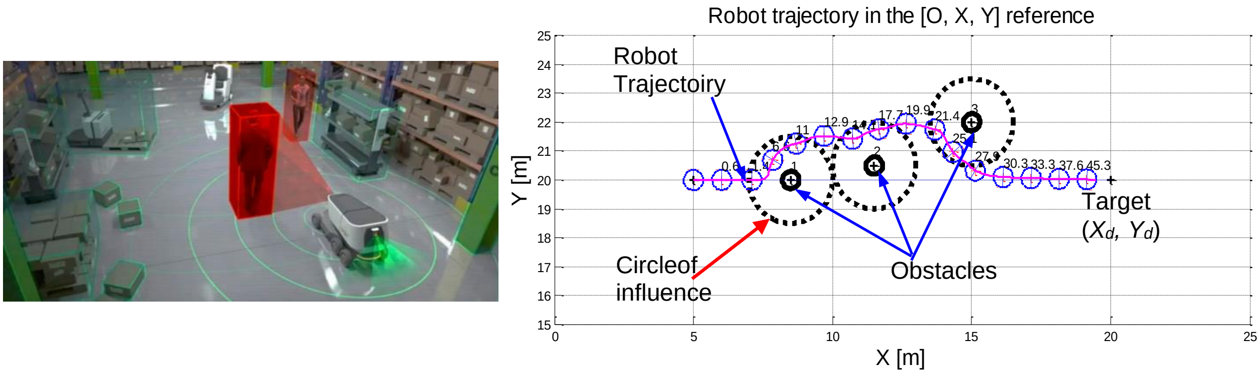
Multi-Controller Architecture (MCA) and Obstacle Avoidance based on Limit-Cycles approach

Keywords: Autonomous navigation, mobile robot modelling, multi-controller architecture, obstacle avoidance, limit cycle method, short and long-term planning, non-linear control, Lyapunov stability, graph theory.

I) Introduction

The main objective of these tutorials is to perform, under MATLAB/Simulink, several kinds of navigations of a unicycle-type mobile robot in a cluttered environment (cf. Fig. 1 (a) and (b)). More specifically, these tutorials address: the modeling of mobile robots; the synthesis of several control laws using Lyapunov stability theorem; the use of the limit cycles method to achieve reactive and cognitive obstacle avoidance (short- and long-term planning respectively); finding the optimal global path based on graph theory.

To carry out these tutorials, you will rely on the one hand on the followed lectures “Planning and control of mobile robots”, and on the other hand on a set of MATLAB / Simulink programs made available, as well as on some papers provided in the “Papers” working directory [1] [2].



(a) (b)

Fig. 1 Navigation of a mobile robot in a cluttered environment: (a) dynamic, (b) static.

II) Tutorial’s assessment

The assessment (which will contribute with 10% of the final mark of SY28 project) will be made on the basis of:

- The level of implication (questions / interactions) of the student during the tutorials/labs.
- An individual small report in pdf version, which summarizes the answers to the questions. You will need to justify your answers using in particular: analytical developments, graphics, tables and image captures linked to the carried-out simulations.
- A folder, containing all the MATLAB / Simulink code used to answer the questions. Do not hesitate to create a specific directory for questions requiring specific explanations, for example to answer question IV.C, name the directory "Response_IV_C".

You must return all the elements related to these tutorials, via a compressed archive one week after the end of the dedicated sessions (I will let you know the actual deadline during the last session of this tutorial’s sub-part). The compressed archive must be sent by email to lounis.adouane@utc.fr.

The evaluation criteria for the final grade of these tutorials are as follows:

- editorial quality of the report (**10%**);
- quality of the obtained results and relevance of the comments made in the report (**40%**);
- quality and clarity of the MATLAB / Simulink produced code (**10%**);
- quality and degree of finalization of the questions marked with "*" (**40%**). Indeed, the questions with "*" are a little bit more complex / tedious to do than the others (in particular requiring more documentation). Thus, feel free to go over all the other questions before coming back to them.

PART I

III) Kinematic model of the robot

a) In section III.B of article [1], the kinematic model of a fixed-point P_t with respect to a unicycle-type mobile robot is given by equation (1). Show that this model is none other than the generalization of the kinematic model of a unicycle given in the course.

b) The kinematic model of the mobile robot is implemented in the *SF_Unicycle.m* function. Look carefully at all the MATLAB / Simulink programs constituting the provided mini-software, specifically *ProgrammePrincipal.m*, *CoordinationControleurs.m*, and the Simulink *ArchitectureControleNavigation.mdl* file (cf. Fig. 2). Write in algorithmic language (in pseudocode) the different steps involved in controlling the mobile robot.

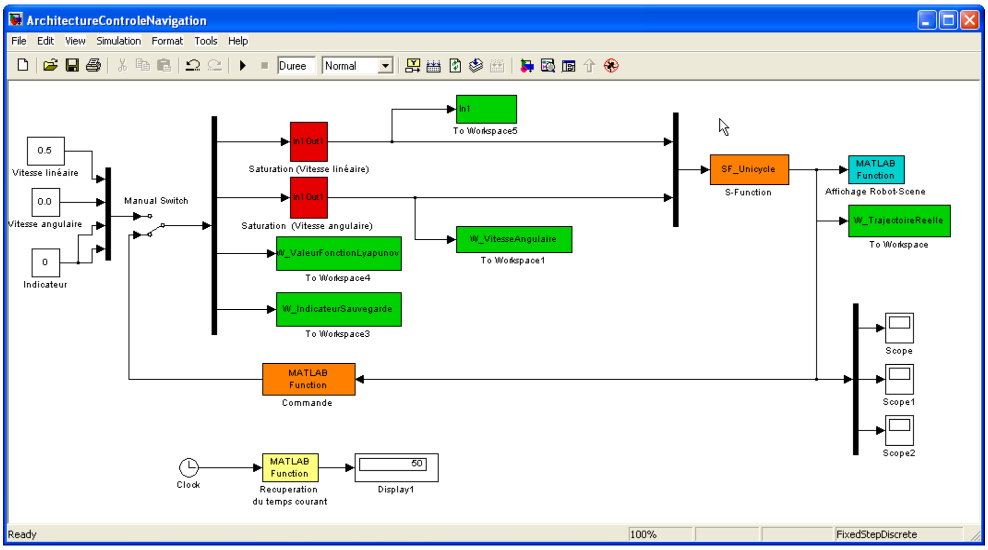


Fig. 2 Simulink scheme used for simulations

IV) Control synthesis using Lyapunov stability theorem

One of the basic tasks that a mobile robot must be able to perform in its environment is to move to a desired position (X_d, Y_d) in a stable and precise manner. This is done using the control law proposed in [1] (cf. equation 3).

a) Show using Lyapunov stability theorem that the proposed control law will guarantee the convergence of errors (e_x and e_y) toward zero.

b) Implement this control law by modifying the *CommandeAttraction.m* function.

c) *

Propose a control law other than the one proposed in [1] and implement it in the file "*CommandAttraction2.m*". You must create this file on the same principle as "*CommandAttraction.m*".

NB: Do not hesitate to draw inspiration from the many works in the scientific literature proposing stable control laws (in the sense of Lyapunov stability) to ensure the convergence of a unicycle robot towards a static target.

V) Obstacle avoidance based on limit-cycle method, and multi-controller architecture

The limit-cycle method (cf. [1] section III.B.2) will be implemented in what follows. Before starting, comment out the « Force activation of the controller "Attraction to the Target" » part of the *CoordinationControleurs.m* function and remove the comments from the « Selection Process » part of the same .m file.

a) To control the robot, it must be precisely aware of the configuration in which it is located, in order to make the most relevant decisions. For these purposes, you must complete and modify the code parts of *VariablesCommandeEvitement.m*, specifically:
   - the equation (12) of the obstacle avoidance algorithm,
   - and equations (5) and (6).

b) Demonstrate using the Lyapunov stability theorem that the proposed control law (9) will guarantee the convergence of the orientation error θ_e towards zero.

c) Implement the orientation control law for following the limit-cycles. To do this, you must modify the *CommandeEvitement.m* function.

d) Test the obtained control algorithm by putting several obstacles in the environment. Comment your approach.

e) What could you say about the stability of the overall multi-controller architecture? Use for your analysis the figure representing the evolution of Lyapunov functions w.r.t. the time.

PART II

VI) Cognitive long-term planning

The autonomous navigation discussed in the previous part is qualified as reactive. It is proposed in what follows that the robot performs Long-Term Planning (LTP, known also as cognitive), based on limit cycles. In fact, in this part of the tutorial, the robot's decisions (e.g., which obstacle to avoid as a priority at time "t", to avoid the obstacle by passing to its left or to its right, ...) must no longer be dependent all times on the configuration of the most annoying / near obstacle, but on the configuration of all the obstacles hindering the robot's progress towards its final target.

The method to be implemented in the following is inspired by [2] (cf. Fig. 3), but while drastically simplifying the used LTP method:
- the robot is a unicycle;
- the environment of navigation is static;
- limit cycles have circular orbits of convergence (circles of influence);
- the reference frame assigned to each obstacle is orthonormal;
- the criteria for characterizing an elementary limit cycle will depend only on: the length of the limit cycle (cf. equation 9 in [2]) and the evolution of the curvature of the limit cycle along its trajectory (cf. equation 10 in [2]);
- once the long-term planning has been obtained (i.e., the optimal order of the obstacles to avoid in order to reach the target, and the directions (clockwise or counter-clockwise) to take to avoid each of these obstacles), the robot must follow the instructions issued by the LTP without having the possibility to change the order and direction of the avoidances.

a) Describe in pseudocode the algorithm giving the main steps characterizing this LTP to reach the final target from any initial configuration of the robot (take inspiration from what is given in [2]). It is important to note that once the graph characterizing the overall set of obtained LTPs, it will be necessary to find the optimal path in the tree structure of this graph.

b) *

Implement the optimal algorithm that you described above under MATLAB / Simulink and test it, at a minimum, on an example of 5 obstacles that hinder the robot's progress towards its target. Comment the made choices (e.g., the discretization step of *Mu* and its chosen min and max values, use of a deterministic or stochastic approach to obtain the graph characterizing the LTP problem, ...).

NB: To simplify your developments, additional functions under MATLAB (e.g., library for creating graphs and obtaining the shortest path, function for calculating the curvature of a path, etc.) are also included to the archive of this tutorial part.

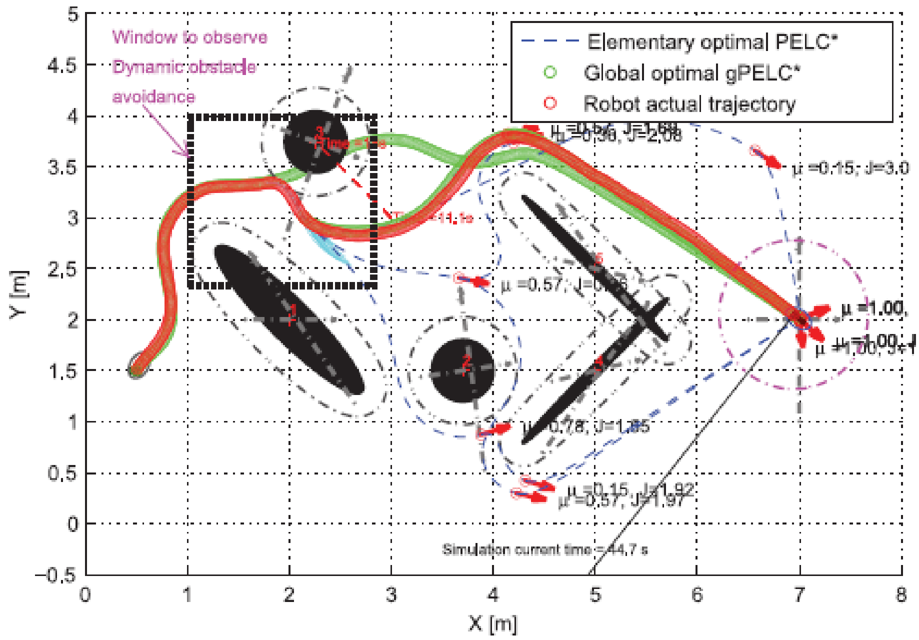


Fig. 3 Cognitive-reactive hybrid navigation based on limit cycles [2]

References

[1], L. Adouane, *Orbital Obstacle Avoidance Algorithm for Reliable and On-Line Mobile Robot Navigation*, Robotica'09, Castelo-Branco Portugal, appeared in Portuguese Journal "Robótica", July 2010.

[2], L. Adouane, *Reactive versus cognitive vehicle navigation based on optimal local and global PELC**. Robotics and Autonomous Systems (RAS), DOI 10.1016/j.robot.2016.11.006, volume 88, pp. 51–70, February 2017.