Computer session on Autonomous Systems: Autonomous navigation and multi-robot coordination 3A ASI - Master MARS

Lara BRIÑÓN ARRANZ

2023-2024

Objectives

The objective of this computer lab is to generate a Matlab code to implement control strategies in order to deal with the following problems:

- autonomous navigation based on artificial potential fields
- consensus-based algorithms for multi-robot coordination
- flocking in multi-robot systems

In the first session (2h), we will design a potential field based strategy to control a robot in order to reach a goal position by navigating through obstacles. In the second session (2h), we will study the cooperative behavior of a multi-robot system which follows a simple consensus control rule. Then, in the last session (2h), both approaches, artificial potential fields and consensus, will be applied to a network of robots to achieve flocking.

Lab evaluation

This computer lab is composed of 3 sessions of 2h each. Groups of two students should be formed to work together during the three sessions.

For the first two sessions, a **written report** with the results (figures and analysis) obtained during the lab is expected for each group. These two reports will be part of the continuous evaluation of your work.

The reports of the first two lab sessions must be **sent at the end of the day of each lab session**, in PDF format, with the names:
BE_Navigation_LASTNAME1_LASTNAME2.pdf
BE_Consensus_LASTNAME1_LASTNAME2.pdf

The last lab session, will be a final test in which each group will produce a report during the lab.

The report of the final session must be **sent at the end of the session** with the name:
BE_Flocking_LASTNAME1_LASTNAME2.pdf

The three reports should be sent to:
lara.brinon-arranz@grenoble-inp.fr
ghadeer.shaaban@grenoble-inp.fr

1 Autonomous navigation with artificial potential fields

In this first exercise, the objective is to design a navigation strategy for a mobile robot to reach a goal position while avoiding obstacles in a two-dimensional space. The strategy to be designed and implemented for the autonomous navigation of the robot will be based on artificial potential fields. Firstly, we will implement a controller for a robot that moves following simple integrator dynamics. Secondly, we will design a new controller based on the same artificial potential field strategy to be applied to a non-holonomic robot.

Consider the following scenario: in a 2-dimensional space with an obstacle, the aim is to drive a robot to a goal position. The robot's position is denoted by $\mathbf{p} = [x, y]^T \in \mathbb{R}^2$, the goal position to be reached by the robot is defined by $\mathbf{p}_{goal} \in \mathbb{R}^2$ and the position of the obstacle is denoted by $\mathbf{p}_{obs} \in \mathbb{R}^2$. To guide the robot toward the goal position while avoiding the obstacle we define an artificial potential field composed of an attractive term and a repulsive one :

$$U(\mathbf{p}) = \alpha U_{attr}(\mathbf{p}, \mathbf{p}_{goal}) + \beta U_{rep}(\mathbf{p}, \mathbf{p}_{obs})$$

where the influence of each term can be set by tuning the parameters α and β.

Question 1: Artificial potential fields strategy (to be prepared BEFORE the first computer lab session)

1.1 Explain briefly the navigation strategy based on artificial potential fields and the role of the attractive and the repulsive potentials. Propose a simple potential field for both the attractive and the repulsive terms. Explain your choice.

1.2 Compute the gradient of your proposed artificial potential field $U(\mathbf{p})$. Explain how this gradient could be used to guide the robot to the goal.

1.3 If we consider that the obstacle is a point without dimensions, the distance vector from the robot's position to the obstacle can be expressed by $\mathbf{p} - \mathbf{p}_{obs}$. However, in real world applications, the obstacles have dimensions and are more than just a point in the space. Assuming that the obstacle has a 2-dimensional circular shape centered at $\mathbf{p}_{obs}$ and with radius R_{obs}, compute the distance vector from the robot position to the nearest point of the obstacle perimeter, denoted by $\mathbf{d}_{obs} \in \mathbb{R}^2$.

1.4 Express the gradient of the proposed artificial potential field $U(\mathbf{p})$ taking into account that the obstacle has a circular shape and using $\mathbf{d}_{obs}$ as described in the previous question.

Question 2: Autonomous navigation for a single integrator robot

First, we consider a robot with single integrator dynamics, i.e., the robot can move in any direction and we can control directly its velocity. The dynamic model of the robot is expressed as:

$$\dot{x} = u_1$$
$$\dot{y} = u_2$$

where $\mathbf{u} = [u_1, u_2]^T$ is the control input.

For this section, use the Matlab code in the file `STUDENTS_Potential_simple_integrator.m` to implement your navigation strategy and controller for the single integrator robot.

2.1 Complete the code of the function `robot_dynamics.m` to implement the single integrator dynamics of the robot.

2.2 To study and analyze the artificial potential field approach in Matlab we have to implement the proposed navigation strategy in discrete time. To do so, the position of the robot at each step must be updated according to its dynamic model. If the sampling time is dt, write the equation to update the position of the robot at each step k using the function `robot_dynamics.m`
Hint: discretize the robot dynamics using $\dot{x}(t) \approx \frac{x(k+1)-x(k)}{dt}$.

2.3 To help you visualize the generated potential field the function `draw_field.m` is provided. Explain how this function works and the relation with the artificial potential field you proposed in Question 1.

2.4 In order to guide the robot to the goal while avoiding an obstacle, design a control law for the single integrator robot by defining a velocity vector reference to be followed, based on the artificial potential field that you proposed in Question 1.

Simulate your system with the following values:

- position of the static obstacle `p_obs=[4;4]`
- position of the goal `p_goal=[6;6]`
- robot's state initial conditions `x=[2;1]`
- potential function parameters `alpha=1` and `beta=1`

Analyze the robot behaviour when the position of the obstacle changes. Illustrate your results with a figure showing the initial and final state of the robot, the trajectory of the robot and the positions of both the goal and the obstacle.

Question 3: Autonomous navigation for a non-holonomic robot

Now, we consider a more complex model for the robot called unicycle. The non-holonomic robot dynamics are expressed as:

$$\begin{aligned}\dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{v} &= u_1 \\ \dot{\theta} &= u_2\end{aligned}$$

where v is the robot's speed, θ its heading angle and u_1, u_2 the control inputs.

For this part of the computer lab, in order to implement your navigation strategy for the unicycle robot complete the Matlab code in the file `STUDENTS_Potential_unicycle.m`. Use the function `draw_robot.m` to draw your unicycle robot.

3.1 The state space of the robot is defined as $\mathbf{x} = [x, y, v, \theta]^T$. Modify the code of the function `robot_dynamics.m` to implement the unicycle dynamics of the robot.

3.2 To study and analyze the artificial potential field approach in Matlab we have to implement the proposed navigation strategy in discrete time. As in the previous case, write the equation to update the position of the robot at each step k using the function `robot_dynamics.m` with sampling time dt.
Hint: discretize the robot dynamics using $\dot{x}(t) \approx \frac{x(k+1)-x(k)}{dt}$.

In order to control the non-holonomic robot the following control strategy is proposed. With the help of the artificial potential field we design a reference velocity vector to be followed by the robot. We will design a proportional controller to make the robot's velocity track this reference velocity. There are two inputs to control the robot, linear acceleration and orientation rate, consequently, we will design two control laws as follows:

$$\begin{aligned}\dot{v} &= u_1 = -K_1(v - v_{ref}) \\ \dot{\theta} &= u_2 = -K_2 \arctan\left(\tan\left(\frac{\theta-\theta_{ref}}{2}\right)\right)\end{aligned} \tag{1}$$

where v_{ref} is the magnitude of the reference velocity vector and θ_{ref} its orientation. For all the simulations, the controller parameters can be set as `K1=1` and `K2=10`.

3.3 In order to guide the robot to the goal while avoiding an obstacle, determine the desired velocity reference to be followed, based on the artificial potential field that you designed in Question 1 and define v_{ref} and θ_{ref} accordingly. Then, code the proportional controller proposed in Eq.(1).

Simulate your system with the following values:

- position of the obstacle `p_obs=[4;4]`
- position of the goal `p_goal=[6;6]`
- robot's state initial conditions `x=[2;1;1;`π`/4]`
- potential functions parameters `alpha=1` and `beta=1`

Analyze the robot behaviour for different values of the initial condition for the orientation θ (simulate for instance `x=[2;1;1;`π`]`). Illustrate your results with a figure showing the initial and final state of the robot, the trajectory of the robot and the positions of both the goal and the obstacle.

3.4 Analyze what happens when the parameter β related to the repulsive potential is 20 times greater than the parameter α related to the attractive potential.

Simulate your system with the following values:

- position of the static obstacle `p_obs=[5;5]`
- position of the goal `p_goal=[6;6]`
- robot's state initial conditions `x=[2;1;1;`π`/4]`

Illustrate your results with a figure showing the initial and final state of the robot, the trajectory of the robot and the positions of both the goal and the obstacle.

Question 4: More realistic scenario with circular shaped obstacles

4.1 Now, we consider a more realistic scenario where the obstacle is a circular shaped object. Following the same reasoning of Question 1.3, complete the code of function `distance_obs.m` in order to compute the distance vector from the robot's position to the nearest point of the obstacle.

4.2 Use the previously defined function `distance_obs.m` to modify the gradient of the potential function with a view to avoid obstacles with a circular shape.

Simulate your system with the following values:

- position of the center of the obstacle `p_obs=[5;5]`
- radius of the circular shaped obstacle `R_obs=0.2`
- position of the goal `p_goal=[6;6]`
- robot's state initial conditions `x=[2;1;1;`π`/4]`

Analyze the robot behaviour for different values of the radius of the circular obstacle. Illustrate your results with a figure showing the initial and final state of the robot, the trajectory of the robot and the positions of both the goal and the circular obstacle.

4.3 Consider now a more complex scenario with two obstacles. The center of the first obstacle is located at $\mathbf{p}_{obs}$ with radius R_{obs} and the center of the second one is located at $\mathbf{p}_{obs2}$ with radius R_{obs2}. Modify the gradient of the potential function to take into account both obstacles.

Simulate your system with the following values:

- position of the centers of both obstacles `p_obs=[4;4]` and `p_obs2=[6;4]`
- radius of the circular shaped obstacles `R_obs=0.2` and `R_obs2=0.2`
- position of the goal `p_goal=[6;6]`
- robot's state initial conditions `x=[2;1;1;`π`/4]`

Analyze the robot behaviour when the radius of the circular obstacles changes. Illustrate your results with a figure showing the initial and final state of the robot, the trajectory of the robot and the positions of the goal and the two obstacles.

2 Consensus for single-integrator robots

The objective of this exercise is to study consensus algorithms applied to a network of single-integrator robots. Consider a group of N robots modeled by the following dynamics:

$$\dot{\mathbf{p}}_i = \mathbf{u}_i, \quad i = 1, ..., N$$

where $\mathbf{p}_i \in \mathbb{R}^2$ denotes the position of robot i and $\mathbf{u}_i \in \mathbb{R}^2$ its control input.

To implement your consensus strategy, use the Matlab code in the file `STUDENTS_Consensus.m`.

Question 1: Consensus protocol under all-to-all communication

For this question, we assume that all the robots are able to communicate with every other robot in the network (all-to-all communication).

1.1 Under this assumption, write and code a compact expression for the Adjacency matrix and the Degree matrix of a network of N robots. What can you say about the graph which represents the connections in the network?

1.2 Write the continuous time consensus protocol for single integrator robots expressed without using the Laplacian matrix. To study and analyze the behavior of the robots in Matlab we have to implement the consensus protocol in discrete time. To do so, the position of robot i at each step must be updated according to its dynamic model. If the sampling time is dt, write the equation to update the position of robot i at each step k.

1.3 Implement a control law for each robot i (without using the Laplacian matrix) to ensure that the network of N robots reaches consensus asymptotically, i.e.,

$$\lim_{t\to\infty} (\mathbf{p}_i(t) - \mathbf{p}_j(t)) = 0 \quad \forall i, j \in \{1, \dots, N\}.$$

Simulate your system for different values of N (for instance, $N = 5$, $N = 50$, $N = 100$) and different initial conditions (use the Matlab function `randn` to get random values). Knowing the initial positions of the robots, can you predict the final consensus position of the group? Explain your answer.
Illustrate your results with two figures:

– a figure showing the initial and final robots' positions and their 2D trajectories.

– a figure showing the robots' positions over time, $\mathbf{p}_i(t)$, for all robots in the network.

1.4 Consider now a network of $N = 5$ robots. Analyze the behavior of the network when a repulsive force among robots is added to the consensus control law. (Hint: Use the simple repulsive potential field studied in the previous exercise dealing with autonomous navigation.)
Illustrate your results with two figures:

– a figure showing the initial and final robots' positions and their 2D trajectories.

– a figure showing the robots' positions over time, $\mathbf{p}_i(t)$, for all robots in the network.

1.5 Express the Laplacian matrix of the network using the previously defined Adjacency and Degree matrices and code a consensus algorithm in a compact form using now the Laplacian matrix. Simulate your system for different values of N. Analyze the convergence rate with respect to N.

Question 2: Convergence analysis of the consensus protocol for different communication graphs

For this question, we assume that the robots are able to communicate only with a subset of robots in the network. The neighbors of each robot are determined by means of a communication graph.

2.1 Consider a network of $N = 4$ robots. Write and code the Laplacian matrix for the four different graphs shown in Fig. 1. What can you say about the properties of these four different graphs (type of graph, connectivity, subgraphs, algebraic connectivity)?

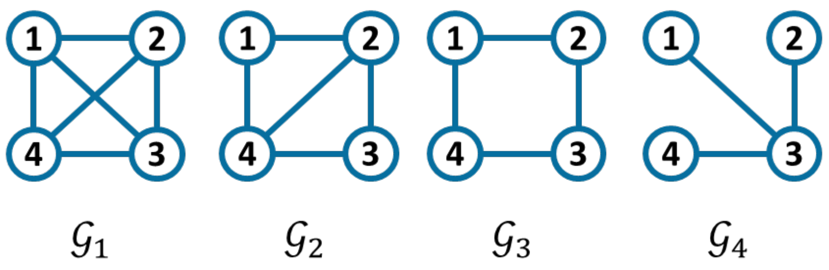

Figure 1: Four different graphs for a network of 4 robots.

2.2 Using the Laplacian matrix, analyze the behavior and convergence of the previously implemented consensus algorithm for the six different network graphs shown in Fig. 1.

Simulate your system with the same values of the initial conditions in order to compare the performances of the six different graphs.

- Can you explain the final positions reached by the robots?
- Is there a mathematical way to determine the convergence rate of the consensus algorithm for each one of the graphs considered?

Compare the convergence rate with a figure showing the robots' positions over time, $\mathbf{p}_i(t)$, for all robots in the network and for each one of the graphs and analyze your results with respect to the algebraic connectivity of each graph.

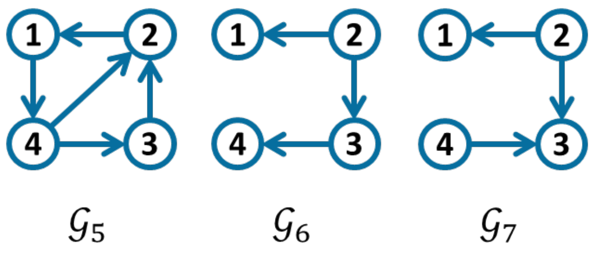

Figure 2: Three different graphs for a network of 4 robots.

2.3 Consider a network of $N = 4$ robots. Write and code the Laplacian matrix for the three different graphs shown in Fig. 2. What can you say about the properties of these three different graphs (type of graph, connectivity, subgraphs, algebraic connectivity)?

2.4 Using the Laplacian matrix, analyze the behavior and convergence of the previously implemented consensus algorithm for the three different network graphs shown in Fig. 2.

Simulate your system with the same initial conditions in order to compare the performances of the three different graphs.

- Can you explain the final positions reached by the robots?
- Is there a mathematical way to determine the convergence rate of the consensus algorithm for each one of the graphs considered?

Compare the convergence rate with a figure showing the robots' positions over time, $\mathbf{p}_i(t)$, for all robots in the network and for each one of the graphs and analyze your results with respect to the algebraic connectivity of each graph.

Question 3: Write a short final conclusion of what you have learned during this lab.

3 Flocking for double-integrator robots

The report and the Matlab code of this final session must be **sent at the end of the session** with the names:
BE_Flocking_LASTNAME1_LASTNAME2.pdf
Code_Flocking_LASTNAME1_LASTNAME2.m

Flocking is a collective behavior that can be found in nature, in schools of fish, flocks of birds and herds of several land animals. In this last exercise, we will combine the previous ideas of potential fields and consensus to implement the three rules of Reynold in order to achieve a flocking behavior in a group of double integrator robots:

R1. Alignment: move towards the average heading of local neighbors

R2. Cohesion: move towards the average position of local neighbors

R3. Separation: move to avoid crowding local neighbors

With this three rules in mind, the objective is to implement a **distributed** control strategy.

Consider a group of N robots with communication capabilities able to transmit their states to their neighbors. Let $\mathbf{p}_i \in \mathbb{R}^2$ denote the position of robot $i = 1, ..., N$, its dynamics are expressed as follows:

$$\dot{\mathbf{p}}_i = \mathbf{v}_i$$
$$\dot{\mathbf{v}}_i = \mathbf{u}_i$$

where $\mathbf{u}_i \in \mathbb{R}^2$ is the control input of robot i.

To implement your flocking strategy, use the Matlab code in the file `STUDENTS_Flocking.m`.

Question 1: Flocking for double integrator robots under all-to-all communication

For this question, we assume that all the robots are able to communicate with every other robot in the network (all-to-all communication).

Question 1.1 To study and analyze the behavior of the robots in Matlab we have to implement the flocking strategy in discrete time. To do so, the state of robot i (position and velocity) at each step must be updated according to its dynamic model. The sampling time is dt and $\mathbf{u}_i = [u_{xi}, u_{yi}]^T$ denotes the control input for robot i.
Write the mathematical equations to update the state of robot i at each step k.

Question 1.2 Design a control law for the behavior R1, i.e., with the objective of aligning the velocity vectors of all the robots, such that:

$$\lim_{t \to \infty} (\mathbf{v}_i(t) - \mathbf{v}_j(t)) = 0 \quad \forall i, j \in \{1, \dots, N\}.$$

Write the mathematical expression of the control law $\mathbf{u}_i$ for each robot i in order to align (R1) its velocity vector with all the robots in the network.

Question 1.3 Code the previous equations in order to simulate your system for different values of N (for instance, $N = 5$ and $N = 10$) and different initial conditions. Draw the robots' positions as well as the robots' velocity vectors.
Illustrate your results with two figures:

- a figure showing the initial state (position and velocity), the trajectory and the final state of all robots in the network,
- a figure showing the velocity vectors over time, $\mathbf{v}_i(t)$, of all robots in the network.

Question 1.4 Analyze the results of the previous simulation. Knowing the initial velocities of the robots, can you predict the final velocity direction of the group? Explain and justify your answer.

Question 1.5 Design an additional term to be added to your previous control law in order to ensure alignment and cohesion behavior (R1+R2) for the group of robots.
Write the mathematical expression of the control law $\mathbf{u}_i$ for each robot i in order to align (R1) its velocity vector with the other robots and to ensure cohesion (R2) among all the robots in the network.

Question 1.6 Code the previous equations and simulate your system for different values of N (for instance, $N = 5$ and $N = 10$) and different initial conditions. Draw the robots' positions as well as the robots' velocity vectors. Illustrate and analyze your results with the help of three figures:

- a figure showing the initial state (position and velocity), the trajectory and the final state of all robots in the network,
- a figure showing the velocity vectors over time, $\mathbf{v}_i(t)$, of all robots in the network,
- a figure showing the positions over time, $\mathbf{p}_i(t)$, of all robots in the network.

Question 1.7 Design an additional term to be added to the previous control law to ensure a flocking behavior (alignment R1, cohesion R2 and separation R3) for the group of robots.
Explain how this separation term works by explaining in detail the approach used and providing the necessary mathematical equations.

Question 1.8 Write the mathematical expression of the control law $\mathbf{u}_i$ for each robot i in order to align (R1) its velocity vector with the other robots, to ensure cohesion (R2) as well as separation (R3) among all the robots in the network.

Question 1.9 Code the previous equations and simulate your system for different values of N (for instance, $N = 5$ and $N = 10$) and different initial conditions. Draw the robots' positions as well as the robots' velocity vectors. Illustrate and analyze your results with the help of three figures:

- a figure showing the initial state (position and velocity), the trajectory and the final state of all robots in the network,
- a figure showing the velocity vectors over time, $\mathbf{v}_i(t)$, of all robots in the network,
- a figure showing the positions over time, $\mathbf{p}_i(t)$, of all robots in the network.

Question 1.10 In order to modify the influence of each term in your control law corresponding to the three behaviors R1, R2, and R3, you would need to add the appropriate parameters. You can use the proposed control parameters `alpha`, `beta`, `gamma` provided in the code. Analyze the influence of these parameters in the behavior of the robots via simulations, in particular, by changing the value of the parameter related to the separation term.

Question 2: Flocking for a distance dependent communication network

For this question, we assume that the robots are only able to communicate with its close neighbors. Let ρ denote the communication radius of the robots, meaning that robot i is able to transmit and get information from the robots that are closer than a distance ρ. In other words, robot i communicates with robot j if $\|\mathbf{p}_i - \mathbf{p}_j\| < \rho$.

Question 2.1 Modify the previous flocking (R1+R2+R3) control law in order to take into account the new distance dependent communication network.
Illustrate and analyze your results with the help of three figures:

- a figure showing the initial state (position and velocity), the trajectory and the final state of all robots in the network.
- a figure showing the velocity vectors over time, $\mathbf{v}_i(t)$, of all robots in the network.
- a figure showing the positions over time, $\mathbf{p}_i(t)$, of all robots in the network.

Question 2.2 Analyze the behavior of the group of robots with respect to the variations on the communication radius ρ.

Question 3 (optional): Flocking with a formation objective function.

For this question, we can assume all-to-all communication in a first time and assume a limited communication radius ρ in a second time. Now, the objective for the group of robots is to achieve flocking (R1+R2+R3) and at the same time, to follow a desired path. The desired path is a circular trajectory of radius R and angular velocity ω. To follow this path, the velocity of each robot i should converge to the reference velocity $\mathbf{v}_{ref}$, such that:

$$\mathbf{v}_i \to \mathbf{v}_{ref} = [-R\omega \sin(\omega t), R\omega \cos(\omega t)]^T, \, \forall i \tag{2}$$

The desired path and the parameters defining it are know for all the robots. To ensure that the formation of robots tracks a desired path, an additional term must be added to the control law of each robot.

Question 3.1 The desired reference velocity have to be implemented in discrete time in Matlab. Write the expression of $\mathbf{v}_{ref}$ defined in Eq.(2) to be computed at each step k knowing that the sampling time is dt.

Question 3.2 Design a simple proportional control term to make each robot i converge to the desired velocity reference $\mathbf{v}_{ref}$.
Write the mathematical expression of this control term.

Question 3.3 Code this additional control term to be added to your previous flocking strategy in order to make the group of robots achieve a flocking behavior and follow the desired circular path. Simulate your system with the following parameters for the reference velocity, $R = 20\,m$ and $\omega = 1\,rad/s$. Illustrate and analyze your results with the help of three figures:

- a figure showing the initial state (position and velocity), the trajectory and the final state of all robots in the network,
- a figure showing the velocity vectors over time, $\mathbf{v}_i(t)$, of all robots in the network,
- a figure showing the positions over time, $\mathbf{p}_i(t)$, of all robots in the network.

Question 4: Write a short final conclusion of what you have learned during this lab.