Robotics

(Lab 2 guidance and workplan sketch)

João Silva Sequeira[1]

[1]joao.silva.sequeira@tecnico.ulisboa.pt

May 9, 2020

TÉCNICO LISBOA

Architecture

1. Draft an architecture for the full system
2. Specify the desired behavior for each building block
3. Specify the lines connecting the blocks and information passing
4. A good diagram of this architecture should be included in the final report
5. Define an interface to input data to your system and to allow visualization of the results
6. Remember to log as much information as possible (sometimes the behavior of the system may not be clear and the logs may help to understand any issues)

Software structure I

- Either Matlab or Python are suitable
- Roughly, the software structure of this type of system involves
  1. A loop, called execution loop, representing a time interval
  2. At each, iteration some (or all) of the components are executed, e.g., sensors are read, the path planner computes something, the trajectory generation computes something, the actuators are fed in with the control variables, etc
  3. Repeat until some terminal condition is met.
- More sophisticated structures (called real-time structures) involve
  1. Setting up a timer, that generates at regular time intervals an interruption
  2. Each time an interruption is detected a callback function (configured by the timer) is called

Software structure II

3. Inside the callback function the adequate variables are set (usually called semaphores)
4. Each semaphore may be associated with one or more of the components
5. Independently of the timer stuff, create an execution loop (similar as above)
6. Each iteration of this loop checks which semaphores are set and only executes the components associated with them

- The simulation has “events” (see the lab statement)
- Associate the events with specific regions of the environment map
- If the car is located in some neighborhood of a position associated with an event check if the event is active and let your system act accordingly

Software structure III

- Explain, in detail, what is the behavior of your car when facing each of these events

Environment map I

- The environment can be represented in multiple forms
    1. As a grid superimposed on an image of IST campus (check the file in the course webpage)
    2. As a collection of polygonal lines, forming cells, that bound the road areas
    3. Recall from the theory classes that the idea is to get a representation for the connectivity of the free space
    4. ...
- The environment is relatively simple, so creating a decomposition of the map from a grid of cell representation shouldn't be a problem
- Creating a decomposition of the map that describes the free space *******

Path planning I

1. Select the path planning strategy

2. The criteria to select the optimal path may involve (among others):
   2.1 The distance between nodes

   2.2 An estimate of the energy necessary to move between any two nodes

3. Don't forget that the car has real dimensions (it is not a single point); the paths must be such that feasible trajectories will be generated

4. Make extensive testing and record examples for the final report

Path planning II

1. The path planning works on a visibility graph (VG)

2. The nodes of the VG can be identified with points inside the cells obtained from the decomposition of the map

3. Any point inside the cell is admissible; however ...

4. ... using points near the border of the cells may have an undesirable effect at the time of generation of trajectory – the interpolated trajectories can easily cross the border of the cells

5. The middle points of the cells are often a good choice

6. Multiple points inside a cell can also be used – The numbers of nodes must be adapted accordingly

TÉCNICO LISBOA

Car simulation

1. Use the car model discussed in the theory classes
2. Use an Euler discretization
3. The output of the simulator is the position and orientation of the car and the steering angle angle
4. Gaussian noise can be added to the outputs of the simulator
5. Car localization is assumed known, up to the Gaussian noise and possible break ups – which are a simplified emulation of possible GPS signal losses and/or software crashes

Trajectory generation

1. Cubic spline along via points
2. The via points need not to be the nodes in the reference path

Paths vs trajectories – Practical considerations

- A path is usually formed by a small number of points; no time is identified with them
- A trajectory is usually formed by much greater number of points and each points is implicitly identified with a precise instant of time

  At time t the trajectory is at point $x(t)$ ...

Car control

1. Upon selection of a control strategy
2. Simulate the car + control, independently of the rest
3. Check carefully the quality of the trajectory – anything that you wouldn't like to see a real car doing it's likely not a good trajectory
   3.1 Check the trajectory in the plane AND the orientation of the car along that trajectory
4. Plot the controls along the trajectory – Large amplitude, jerky, controls are not good – It is unreasonable to have a real autonomous car making violent maneuvers due to violent turning of the steering wheel

Input the reference trajectory to the control system I

- Given the set of points forming the reference trajectory, there are 2 strategies to feed the control system

- The control system operates in an iterative form; each iteration represents a small interval of time during which the controller drives the system with constant controls

- Alternative 1 - Each point is sent to the control system at the corresponding time
  - If the control system is not fast enough it may happen that the point being sent to the control system is too far from the current position, leading to a potential unwanted behavior

  - If the system moves too fast it may happen the the points being sent to the control system are way behind the current position, causing the system to try going backwards

Input the reference trajectory to the control system II

- Alternative 2 - At each iteration of the control loop search for the closest point of the reference trajectory
  - Search for a new point, slightly ahead of the closest point
  - This “look ahead” must be tuned through trial-and-error

Starting a mission

- Assume that the car always starts in a neighborhood of the reference trajectory
- Assume that the initial orientation of the car is close to the direction of movement on top of the reference trajectory

  This avoids having to consider situations where some maneuvering was needed to put the car in the adequate orientation from where to start following the reference trajectory

Energy consumption

- The energy consumption must be monitored along the mission; if it happens that the car goes out of energy the car should stop immediately
- Note that if controls, namely the linear velocity, is too aggressive the energy consumption increases
- P_0 is a constant arbitrarily selected; the purpose is to model the consumption when the car is moving at constant speed

  Other models for the energy consumption can be used, e.g., including a term to model a possible energy consumption when the car is stopped (after all onboard electronics may still be on)

Including energy consumption in low level control I

- The control of the car can be tuned to make use of the information relative to the available energy; two possibilities below

- As example, using the MPC (see the lecture 15 slides), extend the cost function as

$$J = \sum_{i=0}^{N_y} \left(w_i \|y_{ref_{k+i}} - y_{k+i}\|^2 + \Delta E_{k+i}^2 \right) + w_u \sum_{i=0}^{N_u} \|u_{k+i}\|^2$$

  where, u_k are the controls at instant k, y_k, y_{ref_k} are the position of the car and the reference point at instant k, respectively, and ΔE_k is the energy spent during time interval k to $k+1$

  Note that the term $\|u_{k+i}\|^2$ also accounts for energy consumption, though not in a direct form as ΔE_{k+i}^2

- Solve the MPC as usual; the minimization of J will take into account the energy spent

Including energy consumption in low level control II

- Alternatively, include the energy in the low level control
  - Keep track of the energy spent up to the current instant, $E_k = \sum_{i=0}^{k} \Delta E_i$
  - Compute the available energy for the remaining of the trajectory $\Delta E_{budget_k} = E_{budget} - E_k$
  - Assume that the available energy per step is $\Delta E_{budget_k} / N_{steps\ remaining}$
  - Set the maximal linear velocity to maximum value of the solution of the equation $(M\dot{v} + P_0)\, v = \Delta E_{budget_k} / (\Delta t\, N_{steps\ remaining})$

    It can be easily verified that $v = \Delta E_{budget_k} / (\Delta t\, N_{steps\ remaining}) / P_0$ is a solution of the above equation, and hence this value can be used as a conservative estimate for the maximum velocity that allows the car to reach the end of the mission

- Keep checking regularly for updates of this document