Validation of a controller under state constraints

Joris Tillet[1], Luc Jaulin[1], and Fabrice Le Bars[1]

[1]Lab-STICC UMR CNRS 6285, ENSTA Bretagne

Keywords: Trailer control; Interval analysis; Constraints validation.

Introduction

In 1512, the boat named *La Cordelière* sunk in the *Rade de Brest*. Its wreck is still there, on the seabed or more probably under several meters of sediments. As wrecks of this time are rather rare, and the research area is huge, it leads to an interesting challenge.

Since the wreck is buried under the sand, the only sensor which is likely to detect the wreck is a magnetometer, by sensing the magnetic field perturbations of the anchors. Therefore, in order to hope to find the researched boat, a magnetometer should be dragged near to seabed in all the area of research. This is a very long mission and it is intricate to be sure that the sensor is gone everywhere.

This is why *Boatbot* was developed. *Boatbot* is a semi-rigid inflatable boat on which an electric motor was added behind the steering wheel. So, the boat can be regulated in head. Based on this robot, the objective was to develop some algorithms of control such that the magnetometer dragged by the boat properly follows the desired trajectories, while guaranteeing that some constraints are always respected.

Finding a controller

To be sure that the cable cannot be cut by the propellers of the boat, the idea was to put a kayak between the boat and the magnetometer. In this way, near to the propellers there is only a rope which stays at the surface of the water and can even at worst be cut without losing the magnetometer. This experiment can be seen on the picture of *Boatbot* presented in Figure 1.

So the objective here is to find a controller which can control the position of the magnetometer by acting only on the direction of the boat. To address this problem, a good approach is to consider a car with a trailer, and to try to control the trailer. The trailer should follow a vector field, for instance Van der Pol vector field, like in Figure 2.

A robot is represented by its state vector $\mathbf{X}$

$$\mathbf{X} = \begin{pmatrix} x \\ y \\ \theta \\ \theta_r \end{pmatrix},$$

and its evolution function $\mathbf{f}$

$$\dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}, u) = \begin{cases} \cos(\theta) \\ \sin(\theta) \\ u \\ \frac{1}{L_r}\sin(\theta - \theta_r) \end{cases}.$$

The couple of variables (x, y) represents the position of the car, θ its head and θ_r the head of the trailer. L_r is the distance between the car and the trailer (see Figure 3).

It means that a robot is a dynamic system which is modeled by a differential equation. And the job of the controller is to find the input u of the system with respect to some measurements $\mathbf{Y}$ [3].

The error the controller should canceled is the difference between the course of the trailer and the direction given by the vector field [7]. Using feedback linearization method, a controller can be quite easily found and makes the error converge toward zero in few seconds [6].

Figure 1: Picture of *Boatbot* searching for *La Cordelière*.

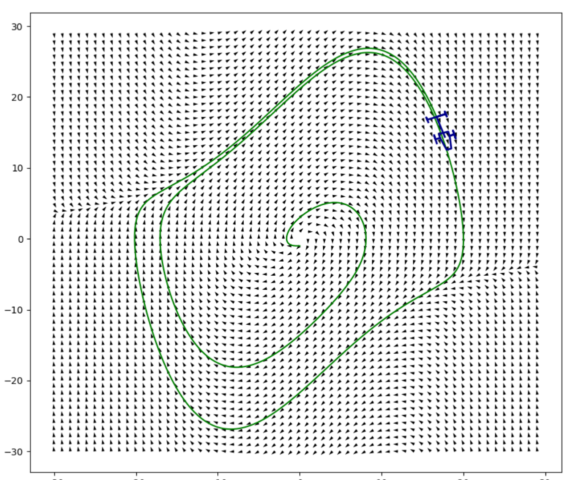

Figure 2: Simulation of a car with a trailer, where the trailer follows the Van der Pol vector field.

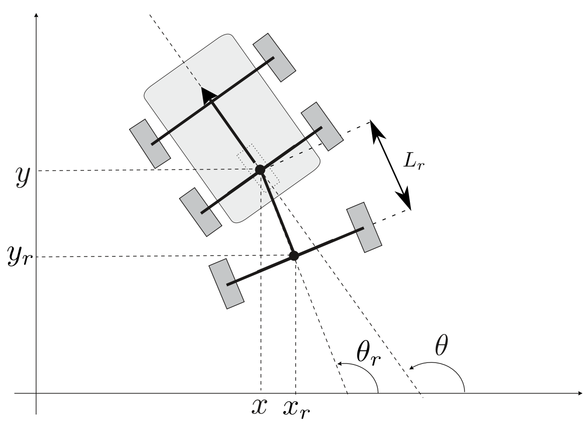

Figure 3: Model of the car with a trailer.

State constraint

Henceforth, the aim is to guarantee that some constraints are always respected. For instance, the angle between the car and the trailer should stay little to ensure that the trailer will never collide with the car itself. So here is presented a mean to show where some constraints are validate or not, supposing the controller works fine. This method relies only on the knowledge of the vector field followed by the robot, using Lie derivatives. This is why there is no need here to integrate any differential equation, like in [1, 2, 9, 10].

As the controller presented hereinabove is supposed to be perfect, we know that the robot will exactly follow the vector field, wherever it is. So the method consists in computing from any position in the vector field the theoretical state of the robot with respect to the controller, and deducing whether the constraint is respected. Interval analysis [8] is used here, helping to find the separator for a given constraint, and to use the SIVIA algorithm [5] to validate the controller in a specific location [4].

An example of result is given in Figure 4. The simulation plots red circles (●) when the constraint is not respected (when there are tight curves), and its trajectory is superposed on the result of the SIVIA: we are sure that the constraint is respected when the robot is in green background areas (■), and violated in orange ones (■). The boundary is in gray, and inside we cannot be sure whether the constraint is respected (■).

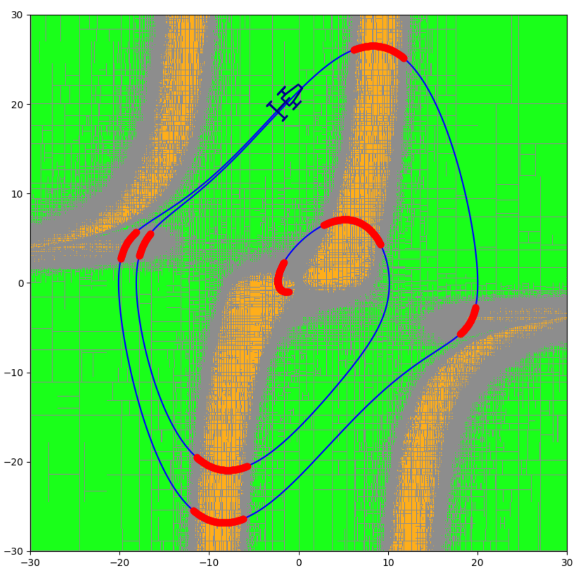

Figure 4: Example of a simulation of the car with a trailer superposed on the SIVIA result. It is certain that the constraint is respected everywhere the background is green (■) supposing the trailer follows the vector field.

Aknowledgement

This work has been supported by the *French Government Defense procurement and technology agency* (DGA). It also benefited from the support of the project CONTREDO of the French National Research Agency (ANR).

References

[1] J. Cruz and P. Barahona. Constraint reasoning with differential equations. *Applied Numerical Analysis & Computational Mathematics*, 1:140 – 154, 03 2004.

[2] M. Janssen, P. V. Hentenryck, and Y. Deville. A constraint satisfaction approach for enclosing solutions to parametric ordinary differential equations. *SIAM Journal on Numerical Analysis*, 40(5):1896–1939, 2002.

[3] L. Jaulin. *Mobile robotics*. ISTE WILEY, 2015.

[4] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis*. Springer-Verlag London, 2001.

[5] L. Jaulin and E. Walter. Set inversion via interval analysis for nonlinear bounded-error estimation. *Automatica*, 29(4):1053–1064, 1993.

[6] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

[7] M. Le Gallic, J. Tillet, F. Le Bars, and L. Jaulin. Tight Slalom Control for Sailboat Robots. In S. M. Schillai and N. Townsend, editors, *Robotic Sailing 2018*, pages 87–93, 2018.

[8] R. E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs N. J., 1966.

[9] S. Rohou, L. Jaulin, M. Mihaylova, F. L. Bars, and S. Veres. Guaranteed Computation of Robots Trajectories. *Robotics and Autonomous Systems*, 93:76–84, 2017.

[10] J. A. D. Sandretto and A. Chapoutot. Dynbex: a differential constraint library-for studying dynamical systems. *Conference on Hybrid Systems: Computation and Control (HSCC2016)*, 2016.