Homework 2 Report

La Scala Giovanni Maria Francesco 241561
Di Lorenzo Andrea 239221

1 Goal of the assignment

The goal of the assignment was to determine the viability kernel for a simple non linear system such as the single pendulum.

The viability kernel is the largest set of feasible states, starting from which the system can always remain inside the set, therefore without violating the constraints. The viability kernel is the largest control invariant set, and its computation has been the subject of many research papers.

1.1 Problem definition

The dynamics of the system composed of a single swinging link connected to a fixed base through a planar revolute joint, could be modeled as:

$$\begin{cases} \dot{q} = v \\ \dot{v} = g \sin q + u \end{cases} \tag{1}$$

The state vector is $x = [q, v]$ and u is the control input. The mass and the length are assumed unitary.

For nonlinear systems, one way in which we can approximate the viability kernel is by solving a sequence of optimal control problems (OCP). We can determine if a state belongs to the viability kernel if there exists a solution of a specific OCP.

The OCP that we need to solve is:

$$\begin{aligned} \text{minimize } & J = 1 \\ \text{subject to: } & x_{k+1} = f(x_k, u_k) \quad \text{for } k = 0, \dots, N-1 \\ & u_k \in U \quad \text{for } k = 0, \dots, N-1 \\ & x_k \in X \quad \text{for } k = 0, \dots, N-1 \\ & x_0 = x_{\text{sample}} \\ & x_N = x_{N-1} \end{aligned} \tag{2}$$

Where $U = [\tau_{\min}, \tau_{\max}]$ and $X = [q_{\min}, q_{\max}] \times [v_{\min}, v_{\max}]$. The final constraint $(x_N = x_{N-1})$ guarantees that the final state of the trajectory is an equilibrium. To implement this constraint we could impose one of the followings:

$$||\dot{q}_N|| < \epsilon_{\text{thr}}$$

$$J(x_N) = \frac{1}{2} w_v (\dot{q}_N)^2$$

The difference in applying one of the previous is that the final cost allows velocity to be non 0 and to stay in the neighborhood of it.

2 Implementing strategy for x_{sample} and generating viability kernel

We ran a series of tests using a grid-based sampling approach that afforded us a systematic exploration of the state space, providing valuable insights into the viability kernel. Our strategy evolved from a simple grid to a more sophisticated one over time, allowing us to grasp the impact of grid granularity on both accuracy and computational efficiency in delineating the viability kernel.

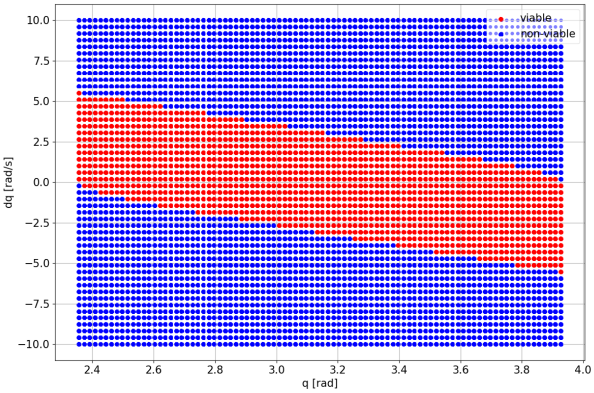

Figure 1: 4500p-g

The implemented algorithm aims to identify the Viability Kernel for the nonlinear system by iteratively solving a series of Optimal Control Problems. Commencing with an initial configuration guess, successful convergence indicates the existence of an optimal solution.

Drawing on Bellman's Optimality Principle, we posit that subsequent states along the trajectory are also optimal. Rooted in dynamic programming, Bellman's principle supports the idea that subarcs of an optimal trajectory maintain their optimality.

Subsequently, starting from a point which admits solution for the OCP, if we sample the states visited during the motion associated to this point we can include them in the viability kernel because thus enlarging our dataset. The results of these tests are shown in `Figure 1`.

We have obtained the same results even with random sampling and the results are shown in `Figure 13`.

3 Considerations on the shape of the viability kernel

The shape of the viability kernel is most likely to be dependent on the system's inherent dynamics and constraints which in this case are the limitations on the maximum and minimum applicable control action and maximum and minimum joint's angle.

At the end of our simulations we got the shape reported in `Figure 1`.

We think that this particular shape is due to the fact that for high velocities a much more aggressive control action is required which may result to a violation on the constraint on the control.

Therefore points with high velocities are discarded from the viability kernel.

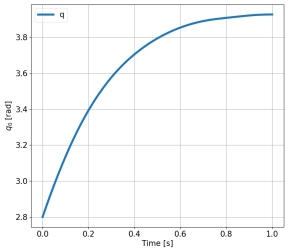

Figure 2: Position

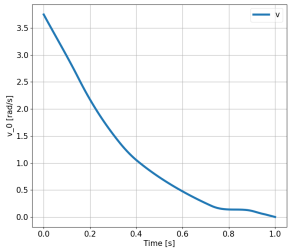

Figure 3: Velocity

If the position bounds are increased (for example up to $\begin{bmatrix}\pi/2 & 3\pi/2\end{bmatrix}$) keeping the velocity bounds to the given ones ($\begin{bmatrix}-10 & 10\end{bmatrix}$), the viability kernel is going to enlarge in height at the ends of the position bounds because the control action manages to stabilize the position even at high velocities as we may see in `Figure 4`.

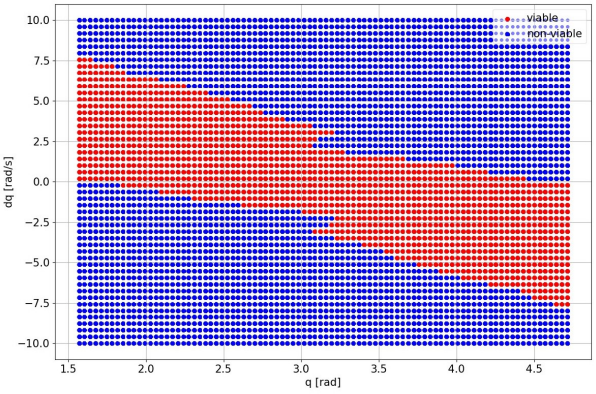

Figure 4: Position augmented to $\begin{bmatrix}\pi/2 & 3\pi/2\end{bmatrix}$

If instead we increase only the velocity bounds, the same considerations about the effort of the controller hold again.

This being said, the overall shape of the viability kernel does not differ much from the one obtained with the original position and velocity bounds.

If the control bounds were to be increased to $5g$, the viability kernel could be approximated to an ellipse reported in `Figure 5`.

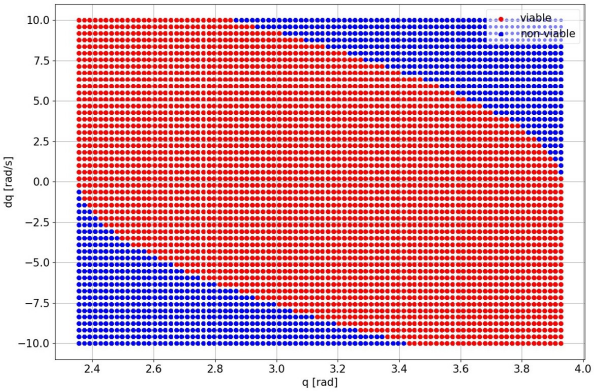

Figure 5: 4500p-5g

As we may see, the control action succeeds in stabilizing the desired position of the pendulum even at high velocities, leading to an increase in dimension of the viability kernel.

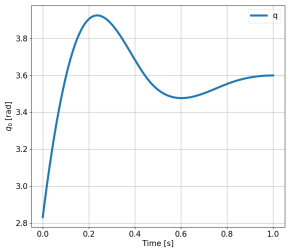

Figure 6: Position

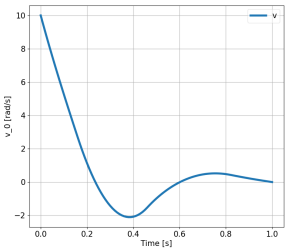

Figure 7: Velocity

The plots above are describing the trajectory of a point near the bound in velocity. As we may see the controller manages to keep the position within the given bounds. The effort in torque, for the control action on this point, is reported in `Figure 12`.

The shape remains the same as in the 1g case. This demonstrates how a more aggressive control action can counteract the effects of velocity, allowing the collection of points at higher velocities.

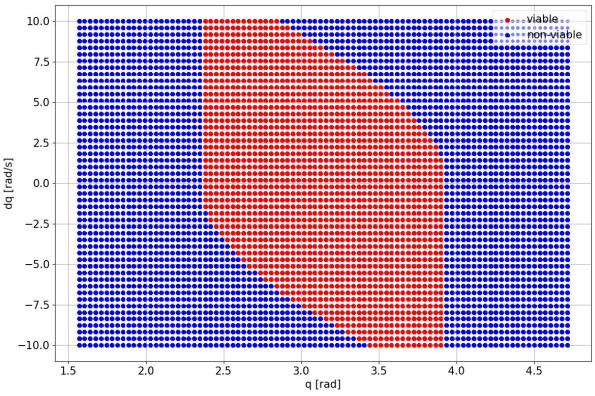

Figure 8: 4500p-5g-2q

The `Figure 8` shows how position and velocity constraints affect a simulation. We created a grid with angles ranging $\begin{bmatrix}\pi/2 & 3\pi/2\end{bmatrix}$ and velocities between $\begin{bmatrix}-10 & 10\end{bmatrix}$. These values were kept within the specified limits. The outcome clearly demonstrates a sharp cut both in position (q) and velocity (dq) values.

This pronounced cut highlights the boundary of the viability kernel, showcasing the region where the system's dynamics comply with the imposed constraints. It is interesting to note how the constraints act as a filter, allowing only a specific set of states to be viable within the defined limits.

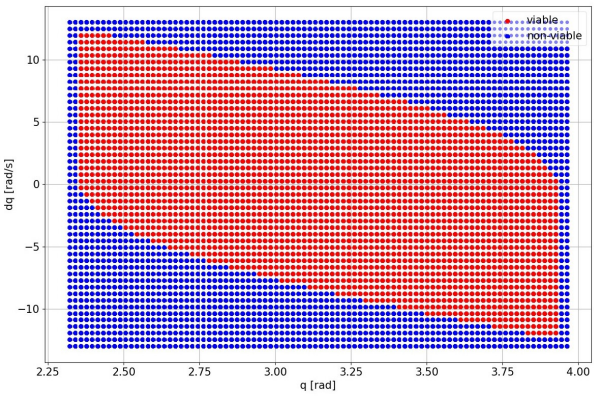

Figure 9: 4500p-5g-vv

The `Figure 9` illustrates the impact of the velocity limit, as anticipated. By increasing both the upper and lower velocity limits, we successfully encompass all data points. It is crucial to note that the overall shape of the figure remains unchanged, but it now includes those points at the extremes that require higher velocities.

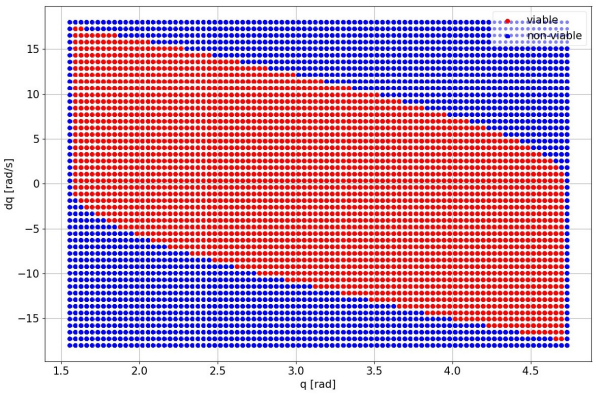

Figure 10: 4500p-2q-2v

The `Figure 10` illustrates the impact of doubling both position and velocity limits while generating a suitable grid (in this case, $\begin{bmatrix}\pi/2 - 1rad & , \pi + 1rad\end{bmatrix}$ and $\begin{bmatrix}-18 & 18\end{bmatrix}$).

Interestingly, increasing the velocity limit has a similar effect as before, while expanding the position limit results in a substantial increase in size without altering the fundamental shape of the figure.

This outcome was somewhat expected since with a sufficiently aggressive control and large limits, the number of valid points significantly increases.

4 Recursive feasibility problem

Recursive feasibility for MPC is solved when all subsequent OCPs are feasible.

In principle, the recursive feasibility would be assured if $N = \infty$ and $l(x, u) \geq \alpha||x||$ for any x, u and $\alpha > 0$ because the time horizon seen by the OCPs would be the same. Therefore a solution would be to increase N.

Our main idea was to fit the data obtained in `Figure 1` and in `Figure 5` (which are a discrete sets) to obtain the analytical expression of the curve. As said we could approximate it to an ellipse whose parametric expression is:

$$\begin{cases} x = x_0 + a\cos\theta \\ y = y_0 + b\sin\theta \end{cases}$$

$$\Rightarrow \frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1$$

The idea is now to impose this equation to be a terminal constraint in our MPC. By definition if we start inside the set limited by the ellipse, we will ensure recursive feasibility. Therefore the problem to be solved will be:

$$\begin{aligned} &\text{minimize } J = 1 \\ &\text{subject to: } x_{k+1} = f(x_k, u_k) \quad \text{for } k = 0, \ldots, N-1 \\ &\qquad u_k \in U \quad \text{for } k = 0, \ldots, N-1 \\ &\qquad x_k \in X \quad \text{for } k = 0, \ldots, N-1 \\ &\qquad x_0 = x_{\text{sample}} \\ &\qquad x_N = x_{N-1} \\ &\qquad \frac{(q - x_0)^2}{a^2} + \frac{(\dot{q} - y_0)^2}{b^2} \leq 1 \end{aligned} \tag{3}$$

Where x_0 and y_0 are the center of the ellipse that we imposed to be the desired equilibrium configuration $[\pi, \quad 0]$.

To implement this idea we saved the viable points in a `.mat` file at run time. Then this file has been parsed to a `MATLAB` script.

The function `fit_ellipse()` returns the value for the major and minor axis of the ellipse (a and b) and the angle of orientation (θ) that best fit the parsed data.

As we may see in figure `Figure 14` to `Figure 15` this is a highly conservative approach but it is enough to ensure recursive feasibility to the MPC in (3). A better alternative to solve this problem is to use a generic fit function (not necessarily an ellipse) and get a polynomial approximation of the viability kernel.

A neural network is a preferable alternative to polynomial fitting due to its ability to handle complex dynamics and adapt to changing system conditions. Unlike polynomials, neural networks excel at learning from simulation data, allowing for more accurate modeling of nonlinear relationships. However, considerations such as computational complexity and the need for a large training dataset must be taken into account to ensure effective generalization across various system conditions.

5 Usefull plots

As shown in `Figure 2` and in `Figure 3` the controller manages to stabilize the system starting at a given joint's angle and velocity even though the torque saturates at the given control bounds (`Figure 11`).

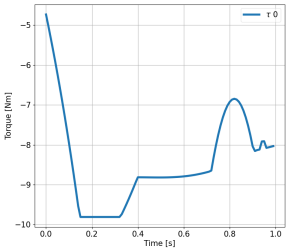

Figure 11: Torque

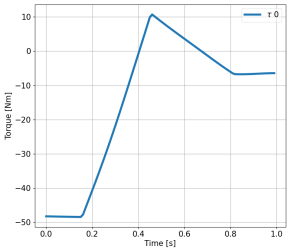

Figure 12: Torque

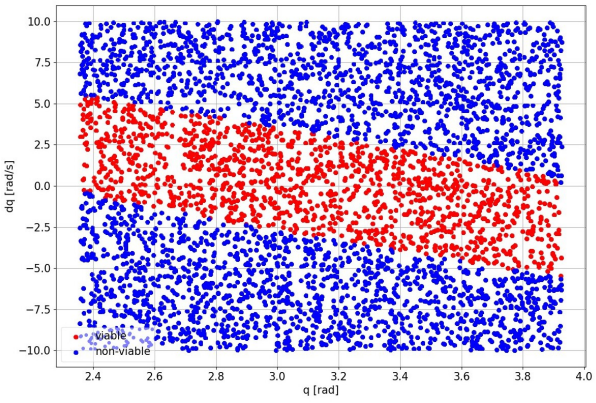

Figure 13: 4500p-1g-rand

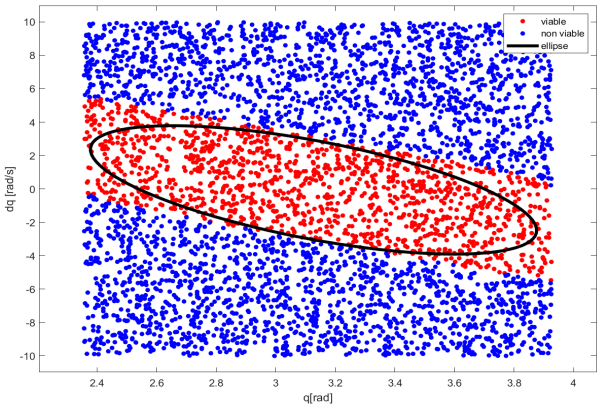

Figure 14: 4500p-ell

Minor axis: $a = 0.5765$ Major axis: $b = 3.8764$

Inclination: $\theta = -0.1252\,rad$ Area: 7.0205

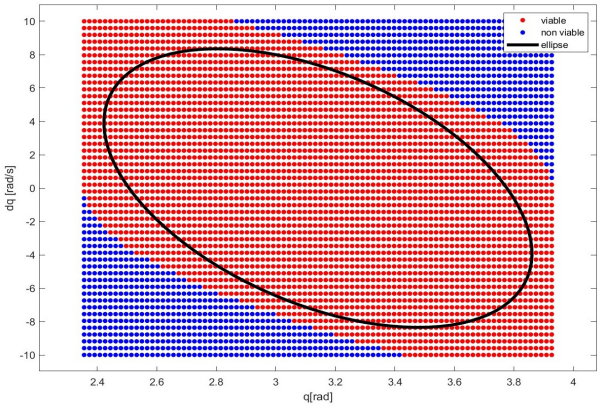

Figure 15: 4500p-5g

Minor axis: $a = 0.6342$ Major axis: $b = 8.6538$

Inclination: $\theta = -0.0407\,rad$ Area: 16.6494