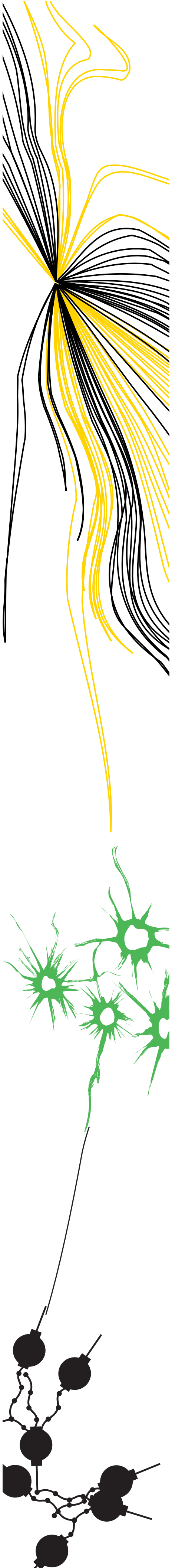


RAM ROBOTICS AND MECHATRONICS

LEARNING FEEDBACK POTENTIAL MAPS USING LARGE-SCALE OPTIMAL CONTROL

R.J. (Roelof Jan) Velthuijs

MSC ASSIGNMENT

Committee:
prof. dr. ir. S. Stramigioli
dr. ir. W. Roozing
dr. ir. F. Califano
dr. ir. A.Q.L. Keemink

January, 2023

002RaM2023
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | TECHMED CENTRE   UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Learning feedback potential maps using large-scale optimal control

Roelof Jan Velthuijs

Abstract— **Impedance controllers are widely used to stabilize feedback systems. Although simple to deploy, these linear controllers are often not optimized for the task at hand. Therefore, a generalized nonlinear feedback law defined as the gradient of the potential and dissipation maps is presented. An optimal control framework is built around the idea that solving a large number of optimal control problems for the same feedback maps will make them optimal for a certain task. The maps can be adapted to a specific scenario by changing the desired task, constraints and objective functions. 2 case studies for 1-DoF positioning tasks are explored and the resulting maps analyzed.**

I. Introduction

In recent years, energy-based control methods have become a point of interest in the control domain. By viewing every system as an entity that store and transforms energy, the energy exchange between two interconnected subsystems can be explicitly analysed in the energy-based framework. Using such a method, a controller, which can be interpreted as a separate system, can be interconnected to the system it needs to control, forming a closed-loop system [1], [2]. The closed-loop system is then controlled by changing the energetic behavior of the controller. One of the most popular controllers which can be studied with an energy-based perspective are impedance controllers. For a mechanical system, an impedance controller uses the power relationship between force and velocity to control a system. A PD-controller that takes a velocity and produces a force is therefore called an impedance controller.

PD-controllers are a common type of feedback controller that have been extensively researched. Some of their advantages are the ease of implementation requiring only two constants and the computational simplicity of finding the optimal constants for linear problems. This makes these controllers well suited for deployment in, e.g. motion tracking or balancing problems, often with the addition of gravity potential compensation [3]. A disadvantage that limits the applicability of PD-controllers is that they can only be optimised for 2 constants, which can lead to sub-optimal behaviour for specific tasks.

As an impedance controller is a PD controller, they have the same advantages and disadvantage. Using a spring and damper constant limits the possible energetic behavior. The optimal behaviour for a controller is dependent on the task, where a task is the objective we desire to achieve. Task examples could be the regulation of an arm and stabilisation of an inverted pendulum.

Variable impedance controllers try to better optimise for a task by changing the impedance through space and/or time. This changes the energetic behaviour throughout the task, which adds a source of possible unstable and unsafe behaviour without additional restrictions on the energetic behaviour of the controller. A few examples of variable impedance controllers with energy budgeting methods are [4]–[7].

The elastic and viscous behaviour of a controller can be generalized by a potential and dissipation map. We can influence the energetic behaviour of the closed-loop system by shaping these maps. This generalization allows us to shape the elastic and viscous behaviour without having to adhere to restrictive predefined shapes, such as the quadratic position dependency of the potential energy of a spring. The optimisation would thus provide more freedom to find a better energetic behaviour for the closed-loop system.

For most controllable systems, a control law that guarantees (asymptotic) stability for the closed-loop system is a desired property. The (asymptotic) stability depends on the shape of the potential and dissipation maps, and shaping such maps is a computationally hard task. Multiple papers have been written on the subject of optimal energy shaping using e.g. neural networks [8], [9].

We propose a new optimisation framework to shape optimal potential and dissipation maps. We optimise the maps by solving the optimal control problem for a finite set of trajectories, where the set of trajectories is defined by a distribution of initial states throughout the system´s state-space and its final states as the desired states. The resulting maps shape the closed-loop system such that it has energetic minima at these desired states, converging to those states from any point within the maps, being (asymptotically) stable around these desired states.

The main contribution of this paper is a framework to compute nonlinear potential and dissipation maps based on a task. This framework consists of a large-scale optimal control problem, which optimizes a potential and dissipation map for multiple trajectories distributed throughout the region of the map.

This paper is organized as follows: Sec. II will provide the technical knowledge required to understand the proposed framework. The control law and a stability proof based on Lyapunov functions are given in section Sec. III. In Sec. IV, we will present the optimisation framework itself. In Sec. V, two case studies will show the optimised controllers, including some simulated results using these controllers. A discussion and conclusion are formulated based on the presented work in Sec. VI.

II. Background

A. Impedance Control

An impedance controller in the mechanical domain is modelled as a spring and damper applying an input force

$$u = k(x_{\text{ref}} - x) - b\dot{x}, \tag{1}$$

where k is the spring constant, b is the damping constant, x is the position, $\dot{x}$ is the velocity relative to the inertial reference frame, and x_{ref} is the desired reference position. By changing x_{ref}, a force is applied that pushes the system back to the reference.

A spring stores potential energy V according to

$$V(x) = \frac{1}{2}k(x_{\text{ref}} - x)^2, \tag{2}$$

the potential function of the spring. Similarly, the damping force can be modelled using the Rayleigh dissipation function

$$R(\dot{x}) = \frac{1}{2}b\dot{x}^2. \tag{3}$$

By summing the negative position and velocity gradients of Eq. (2) and Eq. (3) respectively we obtain

$$u = -\left(\frac{dV(x)}{dx} + \frac{dR(\dot{x})}{d\dot{x}}\right), \tag{4}$$

which is equivalent to Eq. (1).

Optimising the spring and damping constants of an impedance controller has been done, as has optimising them as variable functions over space and time [4]. In this research, we generalize the shape of $V(x)$ and $R(\dot{x})$. Therefore, we will use the terms potential map and dissipation map as generalized terms to describe the elastic and viscous behaviour respectively of the closed-loop system.

Impedance control is limited in how it can shape the energetic behaviour of the closed-loop system. For a constant k, the quadratic dependency on position in Eq. (2) means that the spring force will always be linear. The spring potential only has a single minimum to shape the potential energy of the closed-loop system, whereas some tasks might require multiple minima. This can cause the impedance controller to not be suited for nonlinear systems. For systems with their own potential, e.g. gravity, potential compensation is often used to accompany impedance controllers.

B. Lyapunov Criteria

In this section, we briefly summarise the Lyapunov criteria, which a Lyapunov candidate function should fulfill to qualify as a Lyapunov function. If we can prove that a function is a (strong) Lyapunov function, (asymptotic) stability around an equilibrium can be proven.

Let $\dot{x} = f(x,u)$ be a system, with $\bar{x}$ being an equilibrium of this system. Let $V_{\text{L}}(x)$ be a candidate Lyapunov function and let neighborhood Ω be a subset of the state-space of the system. Let $\bar{x}$ be in Ω and $V_{\text{L}}(x)$ be a function in Ω. If $V_{\text{L}}(x)$ is continuously differentiable, $V_{\text{L}}(x)$ is positive definite defined by

$$V_{\text{L}}(x) > 0, \quad \forall x \in \Omega, x \neq \bar{x}, \quad V_{\text{L}}(\bar{x}) = 0, \tag{5}$$

and $\dot{V}_{\text{L}}(x)$ is negative semi-definite defined by

$$\dot{V}_{\text{L}}(x) = \tfrac{dV_{\text{L}}(x)}{dx}\dot{x} \leq 0, \quad \forall x \in \Omega, x \neq \bar{x}, \quad \dot{V}_{\text{L}}(\bar{x}) = 0, \tag{6}$$

then $\bar{x}$ is a stable equilibrium and $V_{\text{L}}(x)$ is a Lyapunov function. In the case that $\dot{V}_{\text{L}}(x)$ is negative definite defined by

$$\dot{V}_{\text{L}}(x) = \tfrac{dV_{\text{L}}(x)}{dx}\dot{x} < 0, \quad \forall x \in \Omega, x \neq \bar{x}, \quad \dot{V}_{\text{L}}(\bar{x}) = 0, \tag{7}$$

then $\bar{x}$ is an asymptotically stable equilibrium and $V_{\text{L}}(x)$ is a strong Lyapunov function instead. If the closed-loop system admits a Lyapunov function around an equilibrium, the stability of that equilibrium can be proven.

III. Control Law

In this section, we define the potential and dissipation maps and show how they induce a control law for a closed-loop system. A stability proof will be given using Lyapunov analysis and LaSalle's invariance principle.

A. Potential Map

The potential map $V(x)$ is defined as a potential energy storage function that depends on the system configuration x. The negative position gradient of the potential map is a force

$$F_{\text{pot}} = -\frac{dV(x)}{dx}. \tag{8}$$

The potential map is added to other potential energies inside the system, e.g. gravity or stiffness. The total potential energy of the system will then be

$$V_{\text{total}}(x) = V(x) + \hat{V}(x), \tag{9}$$

where V is the potential map and $\hat{V}$ is the potential energy inside the system. An example of this can be found in Fig. 1. The total potential of the system is changed by adding the control potential to the open-loop potential.

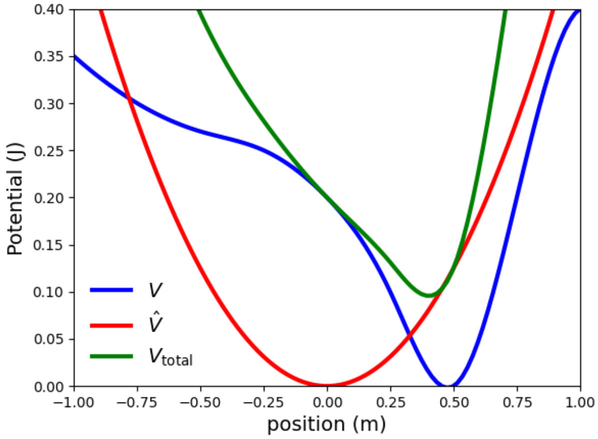


Fig. 1: The blue line is the created potential map, the red line is the potential energy of a system consisting a mass-spring, with $x_{\text{ref}} = 0$ and $k = 1$, such that $\hat{V}(x) = \frac{1}{2}x^2$. For the closed-loop system, they are summed to give V_{total}, placing the energetic minimum of the potential at $x = 0.4$.

B. Dissipation Map

A Rayleigh dissipation function as defined in Eq. (3) is used to model the damping force, were the Rayleigh function is

$$R(x,\dot{x}) = \frac{1}{2}D(x)\dot{x}^2. \tag{10}$$

Here $D(x)$ is a position dependent damping constant, therefore making the damping position dependent. By taking the negative velocity gradient of Eq. (10), we obtain a damping force

$$F_{\text{damp}} = -\frac{dR(x,\dot{x})}{d\dot{x}} = -D(x)\dot{x}. \tag{11}$$

This damping force has a direction opposite to the velocity $\dot{x}$, as long as $D(x) > 0$.

C. Proposed Control Law

The input forces produced by the potential and dissipation map in Eq. (8) and Eq. (11) are summed to compose the control law

$$u = -\Big(\frac{dV(x)}{dx} + \frac{dR(x,\dot{x})}{d\dot{x}}\Big). \tag{12}$$

The closed-loop dynamics then become

$$\dot{x} = f(x,u) = f(x, -\Big(\frac{dV(x)}{dx} + \frac{dR(x,\dot{x})}{d\dot{x}}\Big)). \tag{13}$$

By applying the potential and dissipation maps as input forces within the closed-loop system, an interconnection between the system and the controller is made. The energy of the potential map can be interchanged with the system, and the dissipation map can dissipate energy. The controller changes the behaviour of the closed-loop system, shaping the potential and inserting damping when it is required.

The goal of the feedback law is to regulate the states of the closed-loop system to reach a desired final state x_{f}. The feedback laws should converge any initial state x_0 to the final state x_{f}.

Therefore, we want (asymptotic) stability around x_{f} for the closed-loop system. Next, using a Lyapunov analysis, we prove that the potential and dissipation map as implemented in the closed-loop system Eq. (13) provide (asymptotic) stability around x_{f}, such that they can be used as the feedback law.

D. Stability Proof

We show a stability proof example on a closed-loop system using Lyapunov analysis and LaSalle's invariance principle. This system will be used in Sec. V as a case study. For other systems, a similar analysis can be applied.

We apply the presented strategy in Eq. (12) on a moving mass in 1D to demonstrate an example of closed-loop stability around desired final states. The dynamics of a moving mass are

$$\ddot{x} = \frac{1}{m}u, \tag{14}$$

with $\ddot{x}$ being the acceleration, m being the mass and u an input force. We define Ω as a subset of the state-space. Let the task be to move the system from starting position $x_0 \in \Omega$ to a final position $x_{\text{f}} \in \Omega$, with $\dot{x}_f = 0$. Note that x_{f} needs to be an equilibrium. A common choice for a Lyapunov candidate function is the total energy within the system

$$V_{\text{L}}(x,\dot{x}) = K_{\text{kin}}(\dot{x}) + V_{\text{total}}(x), \tag{15}$$

where the kinetic energy $K_{\text{kin}}(\dot{x}) = \frac{1}{2}m\dot{x}^2$ and $V_{\text{total}}(x)$ is Eq. (9). Since $\hat{V}(x) = 0$,

$$V_{\text{L}}(x,\dot{x}) = \frac{1}{2}m\dot{x}^2 + V(x). \tag{16}$$

Since a Lyapunov function has to be a continuously differentiable positive definite function within a neighborhood Ω of the equilibrium x_{f}, the potential map has to have the same property. Next, we show that $\dot{V}_{\text{L}}$ is negative (semi)-definite.

$$\dot{V}_{\text{L}}(x,\dot{x}) = m\dot{x}\ddot{x} + \frac{dV(x)}{dx}\dot{x}. \tag{17}$$

Using Eq. (14) and Eq. (12), we obtain

$$\dot{V}_{\text{L}}(x,\dot{x}) = -\Big(\frac{dV(x)}{dx} + \frac{dR(x,\dot{x})}{d\dot{x}}\Big)\dot{x} + \frac{dV(x)}{dx}\dot{x}, \tag{18}$$

$$= -\frac{dR(x,\dot{x})}{d\dot{x}}\dot{x}. \tag{19}$$

Using Eq. (10), we get

$$\dot{V}_{\text{L}}(x,\dot{x}) = -D(x)\dot{x}^2. \tag{20}$$

We conclude that $\dot{V}_{\text{L}}$ is negative semi-definite if $D(x) \geq 0$ which implies that the equilibrium is stable. This is not sufficient for asymptotic stability, which would require Eq. (20) to be negative definite.

Using LaSalle's invariance principle, it can be proven that x_{f} is asymptotically stable. let E be a set of the state-space where $\dot{V}_{\text{L}}(x,\dot{x}) = 0$, i.e. $E = (x \in \Omega, \dot{x} = 0)$. Let M be the largest invariant subset of E. To find M in E, $\dot{x} = 0$ for all times. This means that x is constant for all time as well. Eq. (14) then becomes

$$0 = -\frac{1}{m}\frac{dV(x)}{dx} = -\frac{1}{m}\frac{dV(x_{\text{f}})}{dx}, \tag{21}$$

which is satisfied where the gradient is 0, which is at x_{f}. This shows that the system converges towards the largest invariant subset $M = ((x,\dot{x}) = (x_{\text{x}}, 0))$, and we can therefore conclude that the equilibrium x_{f} is asymptotically stable.

This leads to the following constraints for maps defined within a neighborhood Ω to guarantee asymptotic stability:

1) $V(x)$ is continuously differentiable.
2) $V(x)$ is positive definite around its equilibrium x_{f}.
3) $D(x)$ is positive definite.

If the maps are shaped such that they meet these constraints, the equilibrium is asymptotically stable. The third restriction is intuitively understood from Eq. (11): If $D(x) < 0$ then the damping force would have an amplifying effect on velocity, causing unstable behaviour.

For the case of multiple equilibria x_{f} inside Ω, asymptotic stability around those equilibria does not exist within Ω. This is solved by splitting Ω into subsets, where each subset is the neigborhood around an equilibrium such that Lyapunov theorem and LaSalle's invariance principle hold within this subset.

IV. Map Optimisation Framework

In this section, the optimisation framework used to compute the optimal potential and dissipation maps is explained. First, we will introduce the motivation behind the construction of the framework, after which the continuous time problem will be formulated, including the map functions, constraints, and cost functions. Next, the implementation of the transcription is explained. Lastly, we formulate the transcribed nonlinear program (NLP).

A. Concept

The following hypothesis is the motivation behind the framework: Solving a large number of optimal control problems that are functions of the same feedback maps allows for the computation of maps that are optimal for a given task.

We choose a number N_{traj} trajectories, where each optimal control problem consists of regulating a single trajectory to a desired final state using the same feedback maps. We define x_0 as the initial position states and x_{f} as the final position states of a trajectory. For this research we are looking at position regulation tasks, meaning that the initial and final velocity $\dot{x}_0$ and $\dot{x}_f$ are zero. We define Ψ as a region of the state-space. A subset of Ψ, Ω, is then defined as being between positions and having zero velocities. For each trajectory we pick x_0 from a homogeneous distribution within Ω. $x_{\text{f}} \in \Omega$ is chosen to be the same for each trajectory. If enough trajectories are used, this should result in feedback maps that regulate all $x \in \Omega$ to x_{f}.

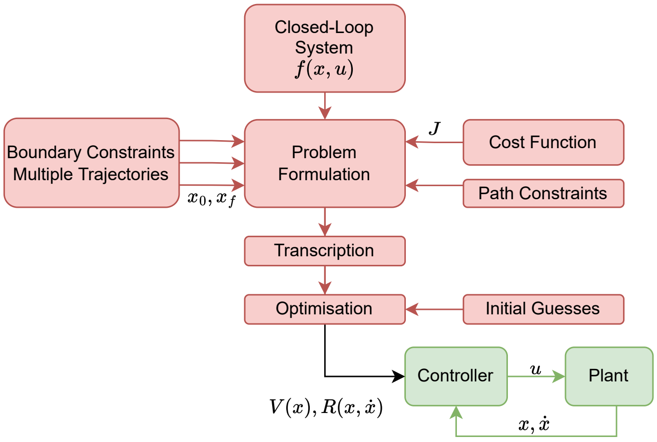


Fig. 2: The optimization framework and controller implementation are depicted in a block diagram. The red blocks are part of the optimisation done offline, whilst the green blocks are the on-line feedback control.

A block diagram of the optimisation framework can be found in Fig. 2. A closed-loop system is described by the system dynamics and the feedback law, consisting of the potential and dissipation maps. The trajectories are initialized as boundary constraints. Further path constraints can be added to specify the task. The cost function can be chosen based on the desired behaviour of the closed-loop system, and initial guesses improve the chance of convergence to an optimal solution. Transforming the continuous time problem into an algebraic one is done using direct collocation, as further explained in Sec. IV-E, resulting in a NLP. The outputs of the optimisation are the optimised potential map $V(x)$ and dissipation map $R(x,\dot{x})$ as well as the optimised trajectories. These optimised maps can be implemented as closed-loop controller.

B. Continuous Time Optimal Control Formulation

In this section, we will formulate the continuous time optimal control problem. To improve readability, we omit the time dependence from the variables whenever it is clear from context. We describe the closed-loop dynamics of a system by $\dot{x} = f(x,u)$, where we use Eq. (12). We want to regulate the system's states $x \in \Omega$ to go to $x_{\text{f}} \in \Omega$. We choose N_{traj} trajectories as defined in Sec. IV-A.

For a time horizon $t \in [0,T]$, the continuous time optimal control problem is formulated as

$$\begin{aligned} \underset{V(\cdot),R(\cdot)}{\text{minimize}} \quad & J(x,u) = \tfrac{1}{N_{\text{traj}}} \textstyle\sum_{i=1}^{N_{\text{traj}}} (L(x_i,u_i) + K(T_i)) + J_r, \\ \text{subject to} \quad & \dot{x} = f(x,u), && t_i \in [0,T_i], \\ & u = -(\tfrac{dV(x)}{dx} + \tfrac{dR(x,\dot{x})}{d\dot{x}}), \\ & (x_i(0), x_i(T_i)) = (x_{0,i}, x_{\text{f}}), && (x_{0,i}, x_{\text{f}}) \in \Omega, \\ & (\dot{x}_i(0), \dot{x}_i(T_i)) = (0,0), \\ & T_i \geq 0, && i = 1,2,.,N_{\text{traj}}, \end{aligned} \tag{22}$$

where index i denotes each trajectory and L and K are the running and final cost computed for each trajectory respectively. Furthermore, J_r is a regularization cost and T_i is the final time of each trajectory. Note that x_{f} is the same final state for each trajectory, defining the task.

C. Potential and Dissipation map

This section describes how the maps are defined for the optimisation.

The potential map $V(x)$ and position dependent damping term $D(x)$ should be continuously differentiable and the optimization should have freedom in shaping these functions. We therefore use uniformly distributed third order B-splines functions for $V(x)$ and $D(x)$, which are shaped using spline weight vectors w_V and w_R respectively. These weight vectors have a size of N_{map} and are used as the decision variables to shape the maps. For the optimal control problem, we will write the functions shaping as $V(x,w_V)$ and $D(x,w_R)$. Further details on B-spline can be found in [10].

The spline is valued only within $x \in \Omega$. Outside of this region the spline and its gradient will be 0.

D. Cost Functions

In this section we present cost function examples that are used for the case studies as well. The presented functions describe a metabolic, performance and regularization cost. The specific task is characterised by a specific choice of cost function.

The final cost term K is an example of a performance cost. This cost function represents the squared final time

$$K(T_i) = T_i^2, \tag{23}$$

with T the final time. As the average of the sum of squared final times is taken, Using the squared value of T results in a penalty for trajectory end times that are much larger than the average. An alternative performance cost function using the not-squared value of T allows for more variance in trajectory end times.

The metabolic cost function is an example of a running cost term L. This cost function represents the average effort to reach the final goal. The metabolic cost is defined as

$$L(x,u) = \int_0^T u^2 \mathrm{d}t. \tag{24}$$

Lastly, a regularization cost J_r is added to the total cost function. This cost function penalises high amplitudes of the position dependent damping term. The regularization cost is defined as

$$J_r = 0.01 \frac{1}{N_{\text{map}}} \sum_{j=1}^{N}{}_{\text{map}} w_R^2, \tag{25}$$

with w_R the spline control points of the dissipation function. This regularization term prevents high damping at positions where the trajectories in the NLP do not have a velocity. By using a factor 0.01, the damping will not be minimized if it compromises performance. Note that this is not necessary for $V(x)$ as we choose $x_{0,i}$ such that all the spline control points $w_{V,j}$ are constrained in achieving the task.

E. Transcription Using Direct Collocation

The continuous-time optimal control problem needs to be transcribed to a NLP for numerical solving. We utilise direct collocation, explained in [11], [12]. Instead of the continuous-time dynamics, the dynamics are only imposed at N_{col} collocation points at times t_k with $k = 0, \dots, N_{\text{col}} - 1$. Between the collocation points the system dynamics are approximated by a polynomial spline, where the spline order depends on the collocation order. The inputs and states then become $x_k = x(t_k)$ and $u_k = u(t_k)$. For readability, $f(x_k, u_k)$ is written as f_k.

This research uses trapezoidal collocation, the lowest order collocation method. It approximates states as quadratic splines and dynamics and control inputs as linear splines. Trapezoidal collocation utilises the trapezoidal rule for integration, i.e.

$$\int_0^T f(x,u) \mathrm{d}\tau \approx \sum_{k=0}^{N_{\text{col}}-1} \frac{1}{2} h (f_{k+1} + f_k), \tag{26}$$

with $h = t_{k+1} - t_k$ being the time between two collocation points. As a result, the continuous time system

$$\int_{t_k}^{t_{k+1}} \dot{x} \, \mathrm{d}t = \int_{t_k}^{t_{k+1}} f(x,u) \mathrm{d}t, \tag{27}$$

can be written as

$$x_{k+1} - x_k = \frac{1}{2} h (f_{k+1} + f_k). \tag{28}$$

The final time T_i of each trajectory is defined as

$$T_i = h_i (N_{\text{col}} - 1). \tag{29}$$

By making T_i a decision variable, the optimisation can change the time between collocation points h_i.

F. Nonlinear Program

By combining the previous sections to transcribe the original continuous time optimal control problem, the NLP can be formulated as

$$
\begin{aligned}
&\underset{w_V, w_R, T_i}{\text{minimize}} \quad J_{\text{total}} = \tfrac{1}{N_{\text{traj}}} \textstyle\sum_{i=1}^{N_{\text{traj}}} (L(x_{k,i}, u_{k,i}) + K(T_i)) + J_r, \\
&\text{subject to} \quad
\begin{array}{ll}
x_{k+1} - x_k = \frac{1}{2}(f_{k+1} + f_k) & k \in [0, N_{\text{col}} - 1], \\
u_k = -(\frac{dV(x_k)}{dx} + \frac{dR(x_k, \dot{x}_k)}{d\dot{x}}), & \\
V(x) = V(x, w_V), & R(x, \dot{x}) = \frac{1}{2} D(x) \dot{x}^2 \\
D(x) = D(x, w_R), & w_R > 0 \\
(x_i(0), x_i(T_i)) = (x_{0,i}, x_{f,i}), & (x_i(0), x_i(T_i)) \in \Omega \\
(\dot{x}_i(0), \dot{x}_i(T_i)) = (0,0), & \\
T_i = h_i (N_{\text{col}} - 1) \geq 0, & i = 1, 2, .., N_{\text{traj}}.
\end{array}
\end{aligned}
\tag{30}
$$

An additional constraint $w_{\mathrm{R}} > 0$ is required, to ensure $D(x) > 0 \forall x$, as elaborated in Sec. III-D. Note that this constraint is over-conservative, as the spline is 0 outside of the area it is defined for. This means that the spline could still be positive even with some negative weight.

V. Case studies

The proposed scheme is implemented through the following two case studies. The first case study involves the regulation of a point mass, and as such does not present natural potentials. The second case study involves the regulation of an inverted pendulum in a gravitational field, and as such involves a plant which presents a natural potential $\hat{V}$.

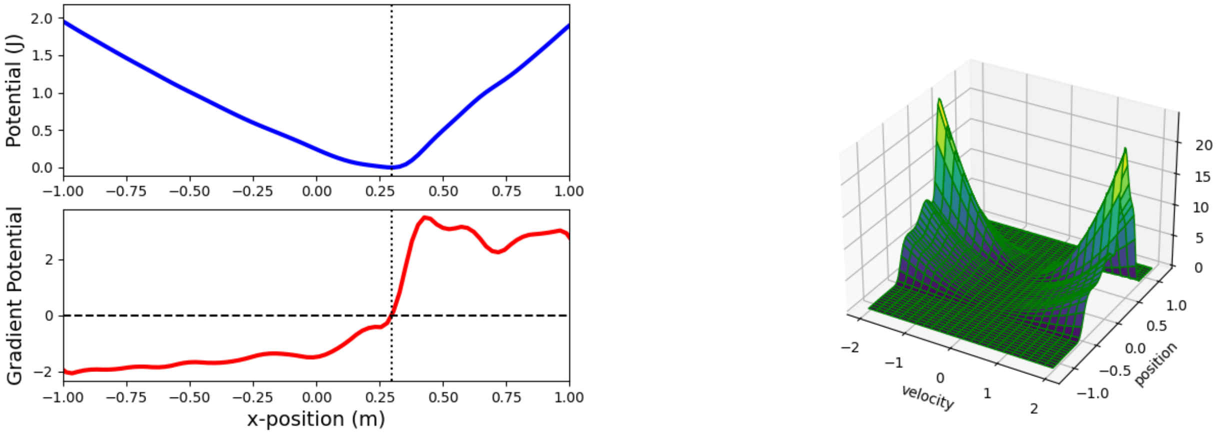
For both case studies, the machine used for all optimisations and simulations is equipped with a AMD Ryzen 5 PRO 5650U CPU and 8 GB of RAM. All optimisation are done using the CasADi[13] open source software in Python. As a solver, IPOPT[14] is used. The function for the optimisable B-spline can be found in [15]. All simulations are done using the odeint function in the SciPy[16] Python package.

A. Case study 1: Shaping a potential function

The first case study involves the optimisation of a nonlinear potential and dissipation map using a point mass.

The differential equation for a 1D moving mass is defined in Eq. (14), in which the input is a force acting on the moving mass. The stability proof for this system was given in Sec. III-D. We use $m = 1\,\text{kg}$.

The NLP as defined in Sec. IV-F is used for implementation in a solver.



(a) The potential map and its gradient. The vertical dotted line indicates $x_f = 0.3$.

(b) The dissipation map

Fig. 3: Case Study 1 - The optimised potential and dissipation maps for the moving mass system.

We define a task, where we want a moving mass position between $-1\,\mathrm{m}$ and $1\,\mathrm{m}$, the set Ω, to converge to $0.3\,\mathrm{m}$. This is formulated for the NLP as

$$\begin{aligned} &\Omega = [-1\,\mathrm{m}, 1\,\mathrm{m}], \\ &x_{0,i} \in \Omega, \qquad\qquad i = 1,2,..,N_{\mathrm{traj}}, \\ &x_{\mathrm{f},i} = 0.3\,\mathrm{m}, \\ &\dot{x}_{0,i} = \dot{x}_{\mathrm{f},i} = 0\,\mathrm{ms}^{-1}. \end{aligned} \tag{31}$$

We provide the solver with initial guesses for the trajectory positions, velocities, and final times. An initial guess for the spline weights of the potential and dissipation maps is not used, meaning the weights will start at 0. As an initial guess for the position for each trajectory, the logistic function is used. This function has an S-shaped curve which resembles the accelerating and decelerating motion the system makes. We therefore define the position initial guess as

$$x_k = x_{0,i} + \frac{L}{1 + e^{-k(t_k - T_i/2)}}, \tag{32}$$

where $k = 8$ is the steepness and $L = x_{\mathrm{f},i} - x_{0,i}$ the amplitude. t_k is the time at each collocation point. If we give an initial guess for the final time, we also have an initial guess for the time at t_k. The final time is guessed as $T_i = 3\,\mathrm{s}$ for all trajectories. The derivative of Eq. (32) is used as initial guess for the velocity,

$$\dot{x}_k = \frac{k \cdot L \cdot e^{-k(t_k - T_i/2)}}{(1 + e^{-k(t_k - T_i/2)})^2}, \tag{33}$$

resulting in a velocity peak halfway through the motion.

The optimisation is done with $N_{\mathrm{col}} = 3$, $N_{\mathrm{map}} = 20$ and $N_{\mathrm{traj}} = 20$.

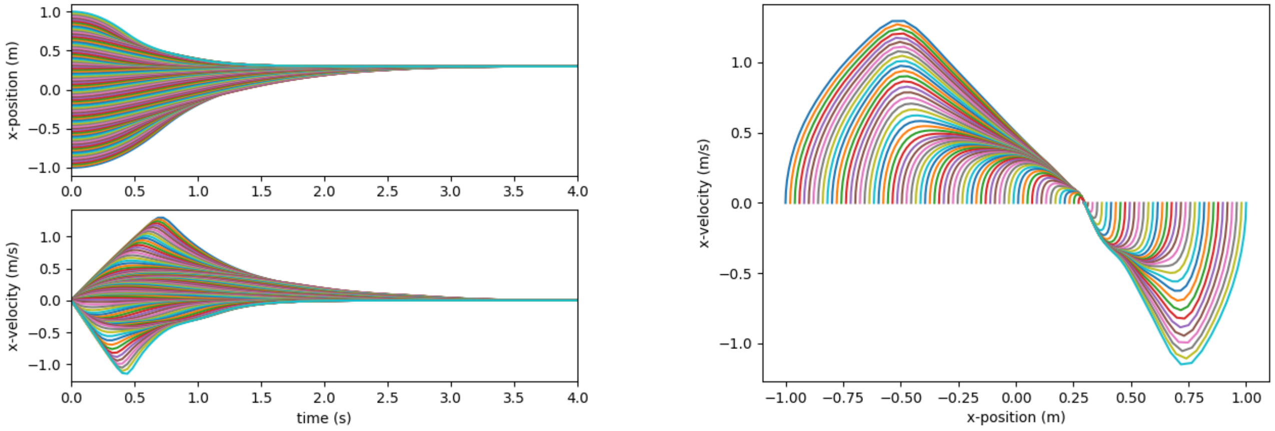
This optimisation problem consist of only 180 decision variables: 20 trajectories with a position and velocity at 3 collocation points and a final time, the potential and dissipation map with 20 spline weights each. This seems like a small problem, but the issue comes from the constraint Jacobian, which has 2080 non-zeros. For the same problem with $N_{\mathrm{col}} = 10$ increases the non-zeros in the constraint Jacobian to 9080. This slows down the speed of the solver significantly. The choice for 3 collocation points will be discussed in the discussion.

As we use a uniform B-spline, the spline weights are equally spaced across the maps. If one of these spline weights does not have a collocation point close to it, then the optimisation can not guarantee a positive definite map anymore. Consequence would be that the maps would not be able to be the feedback law for that part of the map. We want to constrain the potential map spline by having a trajectory start close to every spline weight and therefore we initialize a trajectory for every spline weight.

Cost Functions	Cost
$\frac{1}{N_{\mathrm{traj}}} \sum_{i=1}^{N_{\mathrm{traj}}} K(T_i)$	4.860
$\frac{1}{N_{\mathrm{traj}}} \sum_{i=1}^{N_{\mathrm{traj}}} L(x_{k,i}, u_{k,i})$	3.068
J_r	0.258
J_{total}	8.186

TABLE I: Cost function values for case study 2 after optimisation.

The same hardware and software is used as for case study 1. The IPOPT solver finds a solution after 92 iterations, taking 9.664 s. The cost function values can be found in Table I. The resulting potential and dissipation maps are shown in Fig. 3a and Fig. 3b respectively. The minimum of the potential is at x_{f}, which is the point of lowest energy the closed-loop system should go to. The potential is positive definite around x_{f}, with no other undesired local minima. The gradient of the potential is 0 at x_{f}, even though it does not look very smooth. The position dependent damping term is high at the center of the system, where dissipation is needed to get every trajectory at the minimum, and low at



(a) The state convergence of the moving mass over time.

(b) The phase-space convergence of the moving mass.

Fig. 4: Case Study 1 - Initial value problem of the closed-loop system with the optimised maps.

the outsides where little dissipation is required.

The optimised maps should result in asymptotic stability for the closed-loop system around x_{f}. To verify this, we set up an initial value problem, where 100 trajectories are picked from a homogeneous distribution in Ω. The system is simulated using a time step of $0.04\,\mathrm{s}$. The resulting state-time and phase-space convergence can be found in Fig. 4a and Fig. 4b respectively.

It can be seen that if the position of the closed-loop system lies within Ω, it converges to the desired final position. From this we conclude that the set of optimised maps makes the closed-loop system asymptotically stable around the final state x_{f}.

B. Case study 2: Shaping a nonlinear potential function for a nonlinear system with potential energy.

The second case study involves the optimisation of a potential and dissipation map for an inverted pendulum.

The open-loop system used in this case study is an inverted pendulum with a point mass attached to a massless rod. An actuator in the joint controls the system by applying a counterclockwise positive torque. The second-order differential equation of such a system is

$$\ddot{x} = J^{-1}(\tau_i + mgl\sin(x)), \tag{34}$$

in which $\ddot{x}$ is the angular acceleration, J is the systems inertia, τ_i the input torque, $m = 1\,\mathrm{kg}$ the mass, $g = 9.81\,\mathrm{ms}^{-2}$ the gravitational constant and $l = 1\,\mathrm{m}$ the position of the center of mass of the system.

The inverted pendulum has a nonlinear potential due to gravity

$$\hat{V}(x) = mgl(1+\cos(x)). \tag{35}$$

The total potential energy $V_{\mathrm{total}}(x)$ is then defined as in Eq. (9).

The stability proof in Sec. III-D is altered due to the inclusion of $\hat{V}(x)$. The candidate Lyapunov function, the total energy of the closed-loop system, becomes

$$V_{\mathrm{L}}(x,\dot{x}) = \frac{1}{2}J\dot{x}^2 + mgl(1+\cos(x)) + V(x). \tag{36}$$

The derivative of the candidate Lyapunov function becomes

$$\begin{aligned}
\dot{V}(x) &= J\dot{x}\ddot{x} - mgl\sin(x)\dot{x} + \frac{dV(x)}{dx}\dot{x}, && (37)\\
&= (\tau_i + mgl\sin(x))\dot{x} - mgl\sin(x)\dot{x} + \frac{dV(x)}{dx}\dot{x}, && (38)\\
&= \tau_i\dot{x} + \frac{dV(x)}{dx}\dot{x}, && (39)
\end{aligned}$$

which is the same as Eq. (20).

The NLP as defined in Sec. IV-F is used for implementation in a solver.

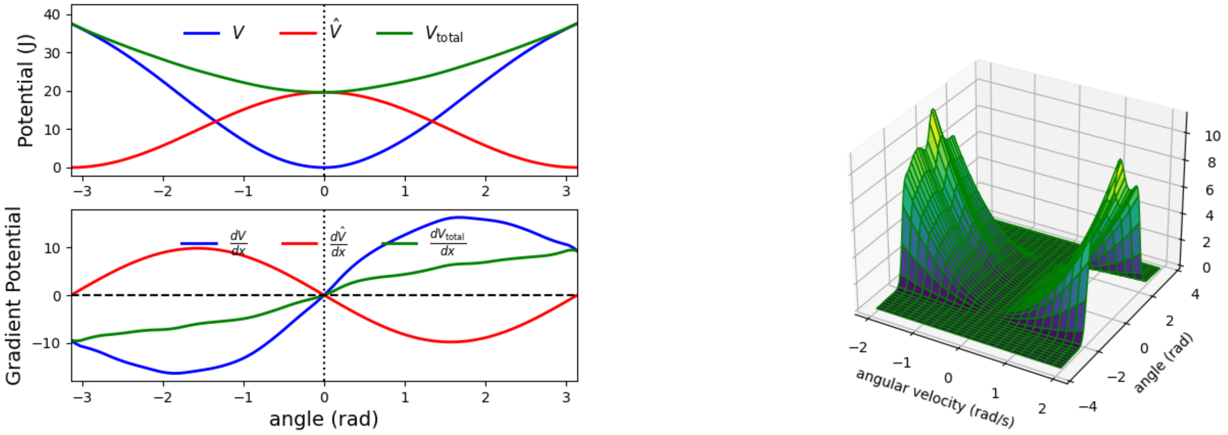
We define a task, where we want the inverted pendulum position between $-\pi$ and πrad, the set Ω, to converge to 0rad. This is formulated for the NLP as

$$\begin{aligned}
&\Omega = [-\pi\mathrm{rad}, \pi\mathrm{rad}], \\
&x_{0,i} \in \Omega, \qquad\qquad i = 1,2,..,N_{\mathrm{traj}}, \\
&x_{\mathrm{f},i} = 0\mathrm{rad}, \\
&\dot{x}_{0,i} = \dot{x}_{\mathrm{f},i} = 0\,\mathrm{rads}^{-1}.
\end{aligned} \tag{40}$$

The same initial guesses for case study 1 are used in case study 2, i.e. Eq. (32) and its derivative Eq. (33). The final time is guessed as $T_i = 3\,\mathrm{s}$ for all trajectories. No initial guess is used for the spline.

The optimisation is done with $N_{\mathrm{col}} = 3$, $N_{\mathrm{map}} = 20$ and $N_{\mathrm{traj}} = 20$, using the same reasoning as in case study 1 for the chosen values.

The same hardware and software is used as for case study 1. The IPOPT solver finds a solution after 171 iterations, taking 18.569 s. The cost function values can be found in Table II. The resulting potential and dissipation maps can be found in Fig. 5a and Fig. 5b respectively. The potential map Fig. 5a also includes $\hat{V}$ and the total potential V_{total}. It can be seen that $V(x)$ modifies V_{total}, placing the energetic



(a) The potential energies and gradients of an inverted pendulum. The vertical dotted line indicates $x_f = 0$.

(b) The dissipation map.

Fig. 5: Case Study 2 - The optimised potential and dissipation maps for the inverted pendulum.

Cost Functions	Cost
$\frac{1}{N_{\text{traj}}} \sum_{i=1}^{N_{\text{traj}}} K(T_i)$	4.11
$\frac{1}{N_{\text{traj}}} \sum_{i=1}^{N_{\text{traj}}} L(x_{k,i}, u_{k,i})$	97.6
J_r	0.122
J_{total}	101.8

TABLE II: Cost function values for case study 2 after optimisation.

minimum of the closed-loop system at x_{f}. The gradient of $V(x)$ has smooth curve, showing that it can compensate for gravity. The dissipation map shows that the position dependent damping term is highest around the equilibrium.

To verify the stability of the optimised maps, we set up an initial value problem similar to case study 1, where 100 trajectories are picked from a homogeneous distribution within Ω. The system is simulated using a time step of 0.04 s. The resulting state-time and phase-space convergence can be found in Fig. 6a and Fig. 6b. It can be seen that if the position of the closed-loop system lies within Ω, it converges to the desired final position. We conclude that the set of optimised maps is able to provide the closed-loop system with asymptotic stability around the final state x_{f}.

VI. Discussion

In this section, possible improvements of the method are discussed regarding the maps, optimal control framework and results.

We generalized the potential and dissipation maps using the argument that more freedom in the map shapes would allow for more possibilities in finding optimal maps for a task.

When comparing impedance controllers with the optimised maps, placing a potential minima is something both can do. The shapes of the optimised maps from the case studies are not of quadratic shape however. The optimisation also takes into account potential energy within the system, therefore not needing potential compensation. For an impedance controller, potential compensation is often used.

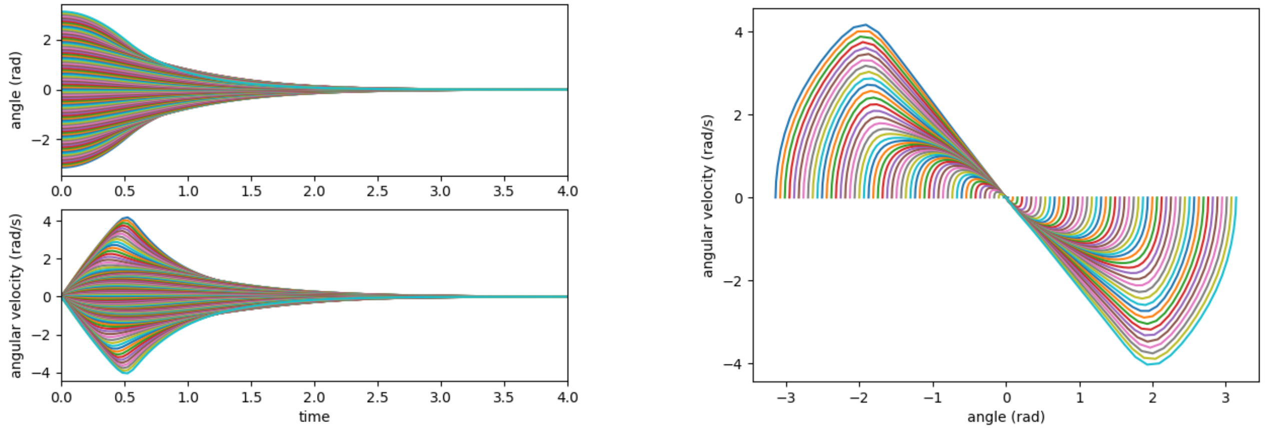
Another difference compared to impedance controllers is the ability to shape multiple minima in a single potential map. A task that would want two minima could be made by letting setting up the optimisation framework to let half of the trajectories converge to one minima and the other half to the second minimum.

The used dissipation map is a position dependent Rayleigh function. For some tasks, it might be interesting to have the $D(x)$ term allowed to become negative. This would lead to local energy generation instead of dissipation, which although unstable, might be useful dependent on the task. Furthermore, in $R(x,\dot{x})$ the velocity term is a quadratic term. Replacing this quadratic term by a function that is positive definite w.r.t. velocity will result in different behaviour for the dissipation function.

The set of position states Ω, from which a linear distribution was picked for the initial boundary constraints has a big influence on the resulting optimised maps. We chose the linear distribution to obtain behaviour that would converge the entire map. Using a normal distribution instead might lead to a larger control action around the equilibrium. The inverse normal distribution could be interesting if we would want behaviour that is mostly focused on trajectories at the edge.

For both case studies, 3 collocation points were used per trajectory as it led to the smoothest map shapes. This however causes for large errors in the dynamics of the optimisation when compared to the simulated results. For more complex systems these errors might cause the problem to become unfeasible, which means that more accurate methods to model the dynamics will have to be implemented.

Path constraints can not reliably be implemented if the



(a) The state convergence over time of an inverted pendulum.

(b) The phase-space convergence of an inverted pendulum.

Fig. 6: Case Study 2 - Initial value problem of the closed-loop system with the optimised maps.

error is too big either. For some tasks, it is relevant that the trajectories are constrained by e.g. a speed limit. Currently, if we set a speed limit, the optimisation might think that it satisfies the speed limit constraint, where in the simulation it goes over this constraint. Reducing the error in the dynamics will allow the usage of path constraints, which gives more options when defining a task.

Regarding the usage of direct collocation, the following improvements can be made. Mesh refinement could be tried to better estimate the trajectories as more collocation points are added, thus imposing the dynamics more.

In both case studies, a second-order system was transcribed using first-order trapezoidal collocation, splitting the system up into a set of first-order equations. In [17], it is stated that transcribing a second order system as a set of first order differential equations leads to errors in the system dynamics. They propose a second-order trapezoidal collocation method for transcribing second-order systems to reduce these dynamics errors. The results show that using second order collocation methods leads to significantly smaller dynamic errors, compared to the first order transcription used in this research. Choosing a higher order collocation method, like Hermite-Simpson collocation, can be another solution to reduce the error in the system dynamics.

One advantage of using 3 collocation points is that it makes the optimisation very fast. This is due to the low-complexity 1D system dynamics used in both case studies. The optimised maps that are obtained in the case studies show asymptotic stability around the equilibrium in the simulations. The compensation of gravity in Fig. 5a could be something for further study, as the system is able to compensate the gravity even though the errors in the optimisation dynamics are significant.

A good initial guess can prevent the solver from getting stuck in local minima. Obtaining good initial guesses can be done by warm-starting the optimisation solver. Warm-starting the solver is done by solving an optimisation problem first and using the results as initial guess for a more complex problem [11]. This could be used to obtain a good initial guess of the spline weights.

The relation between the number of spline weights and the number of trajectories also is of interest. It is clear that when too few trajectories are used for optimisation, the map loses its ability to converge for every state. When too few spline weights are used, it is not optimal to find an optimal shape. Researching the influence of the number of spline weights and the number of trajectories required for a good set of maps could be important when tasks get more complicated.

Lastly, the presented case studies were simple 1D scenarios. Future work consists of applying the optimisation framework to multidimensional systems, e.g. an obstacle avoidance task. Adding dimensions will make the optimisation problem significantly harder to solve and therefore will require better understanding of the framework.

VII. Conclusion

It has been shown that solving a large number of optimal control problems using the same potential and dissipation maps allows for maps that are optimal for a task. This allows for the creation of nonlinear maps and the ability to optimally shape the energy potential inside the closed-loop system.

References

[1] R. Ortega and P. Borja, “Passivity-based control,” in *Encyclopedia of Systems and Control*, J. Baillieul and T. Samad, Eds. London: Springer London, 2019, pp. 1–7. DOI: 10.1007/978-1-4471-5102-9_100072-1.

[2] S. Stramigioli, “Energy-aware robotics,” in *Mathematical Control Theory I*, M. K. Camlibel, A. A. Julius, R. Pasumarthy, and J. M. Scherpen, Eds., Cham: Springer International Publishing, 2015, pp. 37–50.

[3] R. Kelly, “Pd control with desired gravity compensation of robotic manipulators: A review,” *The International Journal of Robotics Research*, vol. 16, no. 5, pp. 660–672, 1997. DOI: 10.1177/027836499701600505.

[4] B. Gerlagh, F. Califano, S. Stramigioli, and W. Roozing, "Energy-aware adaptive impedance control using offline task-based optimization," in *2021 20th International Conference on Advanced Robotics (ICAR)*, 2021, pp. 187–194. DOI: 10.1109/ICAR53236.2021.9659443.

[5] F. Califano, D. van Dijk, and W. Roozing, "A task-based post-impact safety protocol based on energy tanks," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8791–8798, 2022. DOI: 10.1109/LRA.2022.3187254.

[6] G. Raiola, C. A. Cardenas, T. S. Tadele, T. de Vries, and S. Stramigioli, "Development of a safety- and energy-aware impedance controller for collaborative robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1237–1244, 2018. DOI: 10.1109/LRA.2018.2795639.

[7] R. Rashad *et al.*, "Energy aware impedance control of a flying end-effector in the port-hamiltonian framework," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3936–3955, 2022. DOI: 10.1109/TRO.2022.3183532.

[8] S. Massaroli *et al.*, "Optimal energy shaping via neural approximators," *SIAM Journal on Applied Dynamical Systems*, vol. 21, no. 3, pp. 2126–2147, 2022. DOI: 10.1137/21M1414279.

[9] S. Sanchez-Escalonilla, R. Reyes-Báez, and B. Jayawardhana, *Total energy shaping with neural interconnection and damping assignment – passivity based control*, Dec. 2021.

[10] L. Biagiotti and C. Melchiorri, "B b-spline, nurbs and bézier curves," in *Trajectory Planning for Automatic Machines and Robots*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 467–487. DOI: 10.1007/978-3-540-85629-0_11.

[11] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017. DOI: 10.1137/16M1062569.

[12] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, Second Edition*, Second. Society for Industrial and Applied Mathematics, 2010. DOI: 10.1137/1.9780898718577. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898718577.

[13] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019. DOI: 10.1007/s12532-018-0139-4.

[14] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Nature Methods*, 2006. DOI: 10.1007/s10107-004-0559-y.

[15] "Optimisable b-spline function." (), [Online]. Available: https://github.com/casadi/casadi/issues/1484.

[16] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[17] S. Moreno, L. Ros, and E. Celaya, "Collocation methods for second order systems.," *Robotics: Science and Systems Conference. "Proceedings of the XVIII Robotics: Science and Systems Conference (RSS)"*., pp. 1–11, 2022. DOI: 10.15607/RSS.2022.XVIII.038.