IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Error Bounds for PD-controlled Mechanical Systems Under Bounded Disturbances Using Interval Arithmetic

D. Calzolari[1], A. Massimo Giordano[1], and A. Albu-Schäffer[1]

Abstract— **We present a numerical algorithm based on invariant set theory to evaluate the worst-case performance of PD-controlled mechanical systems affected by bounded disturbances. By performing a specific coordinate transformation, the search and computation of positive invariant sets is simplified. It is shown that, thanks to the preservation of problem structure, the proposed method allows to obtain tight, component-wise bounds on the states, which are especially useful for performance evaluation and tuning of a PD controller. The bounds are formally guaranteed and can be used for safety certification. The method is compared to ultimate boundedness, and the superior results are shown via numerical simulations.**

I. Introduction

Performance guarantees are often required in practical control applications. Especially for safety operations, the control system must provide robustness against possible disturbances and uncertainties in order to meet the application requirements. While the analysis and synthesis of robust controllers for linear systems is a very well developed area [1], not the same can be said for nonlinear systems. Control design based on linearized models is not any more state of the art for new generations of torque-controlled, light-weight robots. In order to provide robustness for nonlinear systems, complex controllers have been proposed. Robust control methods, such as sliding mode [2] or model predictive control [3], [4], provide a way to guarantee performance, but these algorithms have other limitations, such as overuse of the actuation [2] or high computational demands [4]. In practice, simple linear feedback controllers (like PD control) can be applied to nonlinear systems and still achieve satisfactory performance, fulfilling the control task requirements [5], [6]. For example, very good practical performance for light-weight robots is achieved by passivity-based impedance controllers with inner loop torque control and friction compensation [7]. While these controllers are used nowadays in many commercial products, theoretical assessment of the disturbed behavior is still limited. Therefore, in this paper, we focus on PD-control, as a template for passivity-based controllers, and we study methods to provide formal and useful error bounds.

Consider a nonlinear system affected by a bounded (possibly non-vanishing) disturbance $w(t)$:

$$\dot{x}(t) = f(x(t), w(t)). \tag{1}$$

Assuming that there exists a Lyapunov function $V(x)$ which proves asymptotic stability in the disturbance-free case, the same function cannot be used if the system is affected by $w(t)$, since it would result $\dot{V}(x, \dot{x}, t) \nleq 0$. Since the disturbance is bounded, the system trajectories most likely remain bounded in a neighborhood of the origin. However, the maximum deviation of the state from the equilibrium cannot be straightforwardly calculated. Usually, methods like Monte Carlo simulations or Rapid Exploring Random Trees (RRTs) are employed to assess the performance of disturbed systems. However, these methods do not to provide formal guarantees on the maximum deviations. Upper bounds on the deviations can be calculated analytically using the ultimate boundedness method [8]. Ultimate boundedness has been exploited to prove robustness and boundedness of families of PD control laws [9]–[12]. Boundedness of the tracking error has also been shown using L_∞ norm analysis [13]. Unfortunately, the aforementioned methods are quite conservative. Because proving ultimate boundedness typically involves norms and Linear Matrix Inequalities (LMI), a certain degree of conservativeness is inherent, mainly due to loss of mathematical properties of the system (e.g., geometric properties, symmetries, and parameter dependencies), also referred to as loss of problem structure [14]. The loss of structure occurs when the worst-case of each term in the dynamics is evaluated separately, and then their effects are combined without taking into account the dependencies between these terms. Usage of norms also leads to the inability to provide component-wise bounds on each state variable, which is useful and required in practice [14]. Even for simple systems (e.g. a 2-DOF planar robot), the estimated error bounds may yield unusable results for tuning purposes, i.e., according to the theory, extremely high gains would be required to meet the precision requirements.

Besides ultimate boundedness, there are also other techniques which exploit certain system properties and geometry by directly manipulating its differential equations, e.g. by associating the original system with a monotone comparison system [15], [16]. However, for multi-DOF mechanical system, manipulation of the equations may be unfeasible.

Numerical methods can preserve local information of a system, without directly dealing with its analytical complexity, at the cost of computational power. These algorithms include non-Lyapunov regression methods [17], reachability analysis [18], [19], and computation of invariant sets for discrete nonlinear system [20]. The discretization of continuous systems [20], bounding of nonlinear terms [17], and further approximations required for computational tractability would lead to unavoidably conservative results. Set-value integration

[1] The authors are with the Chair of Sensor Based Robotic Systems and Intelligent Assistance Systems, Technical University of Munich (TUM), 85748 Garching, Germany, and with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Weßling, Germany davide.calzolari@dlr.de

may provide conservative bounds over long time horizons [18].

In this article we develop a method to guarantee practically useful bounds for PD-controlled nonlinear mechanical systems, or equivalently, for multi-body systems interconnected with passive mechanical springs and dampers. Given a well-tuned controller for convergence of the states, the ultimate objective of this work is to analyze its disturbance behavior. The main contribution of this work is the derivation of an analytical-numerical method to compute non-conservative, formally guaranteed bounds of PD-controlled mechanical systems under the effect of disturbances.

II. Preliminaries

A. Problem statement

Let us consider a n-DOF mechanical system described by

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{K}\tilde{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} = \mathbf{w}(t), \tag{2}$$

where $\mathbf{q} \in \mathbb{R}^n$ is a vector of generalized coordinates, $\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}_d$ is the deviation w.r.t. a desired configuration $\mathbf{q}_d \in \mathbb{R}^n$, $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the mass matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ is the Coriolis/centrifugal matrix, $\mathbf{K} \in \mathbb{R}^{n\times n}$ is a symmetric and positive definite matrix, $\mathbf{D} \in \mathbb{R}^{n\times n}$ is a positive definite matrix, and $\mathbf{w}(t) \in \mathbb{R}^n$ is a bounded disturbance. Functions $\mathbf{w}(t)$ are unknown, but belong to a set $\Omega_{\mathbf{w}}$ of bounded functions. It is assumed that amplitude bounds on the elements of the vector $\mathbf{w}(t)$ are known. Notice that the terms $\mathbf{K}\tilde{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}}$ can result from either passive spring and damper mechanical elements, or active actions enforced by control.

The objective is to analyze the system's behavior in a neighborhood of the operating point $\mathbf{q}_d$. Given (2), we seek at determining nonconservative component-wise bounds on the error $\tilde{\mathbf{q}}$ and on the velocity $\dot{\mathbf{q}}$, under the effect of any realization of the disturbance $\mathbf{w}(t)$. With component-wise bounds we mean that, for some initial state $\mathbf{q}(t_0)$ and $\dot{\mathbf{q}}(t_0)$, it holds

$$|\tilde{q}_i(t)| \leq \alpha_i,\ |\dot{q}_i(t)| \leq \beta_i,\ \forall t \geq t_0,\ \text{ for } i = 1,\ldots,n,$$

where α_i and β_i are scalars.

B. Positive invariant sets

In the attempt to find tight bounds on the state variables, a numerical search of positive invariant sets is performed. Recall the definition of a (robustly) positive invariant set [8], [16]:

Definition 1: A subset $\mathcal{S}$ of the state space of system (1) is said to be robustly positive invariant if for any t_0, $w(t) \in \Omega_{\mathbf{w}}$ and every initial condition $x(t_0) = x_0 \in \mathcal{S}$, the corresponding solution trajectory $x(t, x_0)$ remains in $\mathcal{S}$, for all $t \geq t_0$.

Intuitively, this means that if the system trajectories enter a positive invariant set, they will never escape it. Bounds on the state variables can be then inferred as the boundary of such a set.

C. State Transformation

In the following, a transformation of the state is employed, which facilitates the search of positive invariant sets for system (2). Let us define the following coordinates transformation

$$\mathbf{\Phi}(\tilde{\mathbf{q}}) = \frac{1}{\sqrt{2}}\mathbf{K}^{\frac{1}{2}}\tilde{\mathbf{q}}, \quad \mathbf{\Psi}(\mathbf{q},\dot{\mathbf{q}}) = \frac{1}{\sqrt{2}}\mathbf{M}(\mathbf{q})^{\frac{1}{2}}\dot{\mathbf{q}}, \tag{3}$$

where $\mathbf{K}^{\frac{1}{2}}$ and $\mathbf{M}(\mathbf{q})^{\frac{1}{2}}$ are the square roots of the stiffness and mass matrices, respectively. The new coordinates $\mathbf{\Phi} \in \mathbb{R}^n$ and $\mathbf{\Psi} \in \mathbb{R}^n$ have the property that their squared norms represent the potential and the kinetic energies of the system, respectively. Note that all components of $\mathbf{\Phi}$ and $\mathbf{\Psi}$ share the same unit $\sqrt{J}$ (joule), which plays an important role in scaling quantities, and, at the same time, the contribution of each variable is implicitly tracked. In fact, if bounds on the components of $\mathbf{\Phi}$ are found, then component-wise bounds on $\tilde{\mathbf{q}}$ can be directly calculated.

Without loss of generality, we define $\mathbf{q}_d = 0$, such that $\tilde{\mathbf{q}} = \mathbf{q}$. System (2) can be re-written in the new state $\boldsymbol{\chi} = [\mathbf{\Phi}^T, \mathbf{\Psi}^T]^T$ by using (3) and the inverse transformation as

$$\dot{\boldsymbol{\chi}} = \mathbf{A}(\mathbf{\Phi},\mathbf{\Psi})\boldsymbol{\chi} + \mathbf{G}(\mathbf{\Phi})\mathbf{w}, \tag{4}$$

where the system matrix $\mathbf{A}(\mathbf{\Phi},\mathbf{\Psi}) \in \mathbb{R}^{2n\times 2n}$ and input matrix $\mathbf{G}(\mathbf{\Phi}) \in \mathbb{R}^{2n}$ are expressed by

$$\mathbf{A}(\mathbf{\Phi},\mathbf{\Psi}) = \begin{bmatrix} \mathbf{0} & \mathbf{K}^{\frac{1}{2}}\mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}} \\ -\mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}}\mathbf{K}^{\frac{1}{2}} & -\mathbf{S}(\mathbf{\Phi},\mathbf{\Psi}) - \mathbf{D}_{\mathbf{M}}(\mathbf{\Phi}) \end{bmatrix},$$

$$\mathbf{G}(\mathbf{\Phi}) = \begin{bmatrix} \mathbf{0} \\ \frac{1}{\sqrt{2}}\mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}} \end{bmatrix},$$

with

$$\mathbf{D}_{\mathbf{M}}(\mathbf{\Phi}) = \mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}}\mathbf{D}\mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}},$$
$$\mathbf{S}(\mathbf{\Phi},\mathbf{\Psi}) = \mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}}\mathbf{N}(\mathbf{\Phi},\mathbf{\Psi})\mathbf{M}(\mathbf{\Phi})^{-\frac{1}{2}},$$

where the matrix $\mathbf{N}(\mathbf{\Phi},\mathbf{\Psi}) \in \mathbb{R}^{n\times n}$ is given using vectorized notation and dropping the functional relation as follows

$$\text{vec}(\mathbf{N}) = (\mathbf{I}_n \otimes \mathbf{M}^{\frac{1}{2}})(\mathbf{M}^{\frac{1}{2}} \oplus \mathbf{M}^{\frac{1}{2}})^{-1}\text{vec}(\mathbf{C}^T + \mathbf{C}) - \text{vec}(\mathbf{C}),$$

where "$\otimes$" and "$\oplus$" are the Kronecker product and sum, respectively, "vec($\cdot$)" is the vectorization operator, and $\mathbf{C}$ is parametrized using Christoffel symbols.

Let us define a scalar quadratic function $V : \mathbb{R}^{2n} \to \mathbb{R}$ as

$$V(\boldsymbol{\chi}) = \boldsymbol{\chi}^T\mathbf{P}\boldsymbol{\chi}, \tag{5}$$

where $\mathbf{P} \in \mathbb{R}^{2n\times 2n}$ is a non-diagonal, positive definite matrix. The so called sub-level set $S_{P,c}$ of $V(\boldsymbol{\chi})$ is defined as

$$\mathcal{S}_{P,c} \triangleq \{\boldsymbol{\chi} \in \mathbb{R}^{2n} \mid V(\boldsymbol{\chi}) = \boldsymbol{\chi}^T\mathbf{P}\boldsymbol{\chi} \leq c\}.$$

A level set of $V(\boldsymbol{\chi})$ represents a $2n$-dimensional ellipsoid in space $\boldsymbol{\chi}$. Being $V(\boldsymbol{\chi})$ a quadratic function, assuring that the derivative $\dot{V}(\boldsymbol{\chi},t)$ is negative semi-definite on the boundary set $\partial\mathcal{S}_{P,c} = \{\boldsymbol{\chi} \in \mathbb{R}^{2n} \mid V(\boldsymbol{\chi}) = c\}$ implies positive invariance of $\mathcal{S}_{P,c}$ (tangentiality condition [8], [21]). The derivative of $V(\boldsymbol{\chi})$ along the system trajectories of (4) can be written as

$$\dot{V}(\boldsymbol{\chi},t) = \boldsymbol{\chi}^T\left(\mathbf{P}\mathbf{A}(\boldsymbol{\chi}) + \mathbf{A}(\boldsymbol{\chi})^T\mathbf{P}\right)\boldsymbol{\chi} + 2\boldsymbol{\chi}^T\mathbf{P}\mathbf{G}(\boldsymbol{\chi})\mathbf{w}(t). \tag{6}$$

In general, it is not trivial to analyze the sign of $\dot{V}$ for points belonging to $\partial S_{P,c}$. Even for a 1-DOF nonlinear system, the introduction of disturbances will make it impossible to find an analytical expression of an hyper-boundary $\dot{V}(\boldsymbol{\chi},t) = 0$ in a general case.

D. Interval Analysis

To avoid norms and LMI when dealing with the nonlinearities, Interval Arithmetic (IA) can be used to determine the extreme values of $\dot{V}(\boldsymbol{\chi}, t) = 0$. IA is a model for self-validated numerical analysis, which guarantees bounds on operations involving uncertain terms. In IA, each uncertain variable is modeled with an interval, which represents the set of its attainable values. Formally, we define an interval as the set

$$[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$$

where a is the infimum and b is the supremum. IA has the correctness property [22]: arithmetic operations (like addition, multiplication, etc.) can be defined mathematically for intervals such that the evaluation of an expression always yields an interval containing all results. In the context of this work, the unknown disturbance is modeled as an interval, and IA is used to evaluate the supremum of $\dot{V}(\boldsymbol{\chi}, t)$. For an introduction to IA, please refer to [22]. In the scope of this article, we denote an interval quantity with the superscript $(\cdot)^I$.

Henceforth, the functional relation in the equations will be dropped out, except in the cases where it will be explicitly needed for the analysis.

III. Evaluation of bounds in $\boldsymbol{\chi}$ space

The proposed method consists of the following main steps:

A. Initialize a representation of the unitary $2n$-hypersphere by means of lists of polar angles and vertices.
B. Find a candidate set $\mathcal{S}_{P,c}$ by optimizing $\mathbf{P}$ and c to minimize the bound on a component of $\boldsymbol{\chi}$.
C. Prove positive invariance of the candidate set $\mathcal{S}_{P,c}$.
D. Retrieve bounds in the original system coordinates.

In the next paragraphs, the algorithms required to perform the aforementioned steps are presented.

A. Interval representation of the boundary $\partial\mathcal{S}_{P,c}$

Assuming a region $\mathcal{S}_{P,c}$ is candidate to prove boundedness of trajectories, a coordinate transformation $\boldsymbol{\chi} = \mathbf{Tz}$ can be carried out such that

$$V(\boldsymbol{\chi}) = \boldsymbol{\chi}^T \mathbf{P} \boldsymbol{\chi} = c, \tag{7}$$

$$\boldsymbol{\chi} = \mathbf{Tz} \;\Rightarrow\; V_z(\mathbf{z}) = \mathbf{z}^T \mathbf{z} = 1, \tag{8}$$

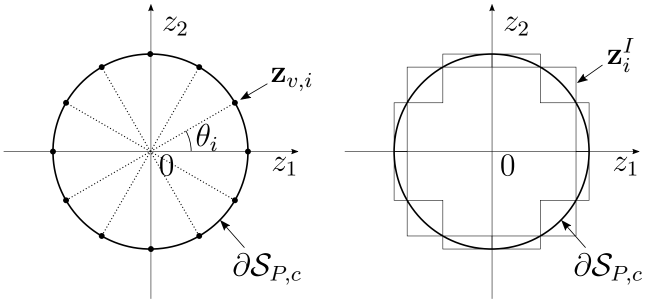
where $\mathbf{T} \in \mathbb{R}^{2n \times 2n}$. A possible choice for $\mathbf{T}$ is

$$\mathbf{T} = \sqrt{c} \cdot \mathbf{V}_P \mathbf{E}_P^{-\frac{1}{2}}, \tag{9}$$

where $\mathbf{V}_P$ and $\mathbf{E}_P$ are matrices containing the eigenvectors and eigenvalues of $\mathbf{P}$, respectively. Substituting (8) in (6):

$$\begin{aligned}\dot{V}_z(\mathbf{z}, t) = \mathbf{z}^T \mathbf{T}^T \left(\mathbf{PA}(\mathbf{Tz}) + \mathbf{A}(\mathbf{Tz})^T \mathbf{P}\right) \mathbf{Tz} \\ + 2\, \mathbf{z}^T \mathbf{T}^T \mathbf{PG}(\mathbf{Tz}) \mathbf{w}(t).\end{aligned} \tag{10}$$

Carrying out the analysis of $\partial S_{P,c}$ using V_z means then to evaluate the sign of $\dot{V}_z$ for the points on an hyper-sphere of radius 1 (see (8)). The evaluation of the sign of (10) is carried out by means of Interval Arithmetic. The range of possible values for the disturbance $\mathbf{w}(t)$ is modeled by an interval $\mathbf{w}^I$. Thus, given a vector $\mathbf{z}$, evaluating (10) results into the interval $\dot{V}_z^I(\mathbf{z}, \mathbf{w}^I)$, where the dependency on the bounds $\mathbf{w}^I$ is made explicit, while that on time is dropped. The usage of the interval representation of the disturbance is an essential step in order to guarantee that the interval $\dot{V}_z^I(\mathbf{z}, \mathbf{w}^I)$ will provide the set of attainable values of $\dot{V}_z(\mathbf{z}, t)$ for any disturbance realization $\mathbf{w}(t) \in \Omega_{\mathbf{w}}$.



(a) polar angle θ_i and vertex $\mathbf{z}_{v,i}$ (b) interval boxes envelope

Fig. 1. Visualization of Algorithm 1, presented in section III-A. Firstly, equi-distributed vertices are generated on the hypersphere (in this case a circle). Each vertex $\mathbf{z}_{v,i}$ is associated with the corresponding polar angles θ_i. Subsequently, interval boxes $\mathbf{z}_i^I$ are used to envelope the boundary.

In order to apply IA on the boundary, an interval representation of it is required. Therefore, the unitary hypersphere (8) is enveloped using $2n$-dimensional interval boxes (see Fig. 1). To this end, a number N of equi-distributed vertices are generated on the unitary hypersphere embedded in the $2n$-dimensional space. This can be performed by extending the algorithm presented in [23] to more than 3 dimensions. The algorithm outputs the equi-distributed vertices $\mathbf{z}_v \in \mathbb{R}^{2n}$ in Cartesian coordinates and a list $\mathcal{L}_{\boldsymbol{\theta}}$ of corresponding polar angles $\boldsymbol{\theta} \in \mathbb{R}^{2n-1}$ to generate them. The vertices $\mathbf{z}_v$ are stored in the list $\mathcal{L}_{\mathbf{z}}^v$. The list of polar angles $\mathcal{L}_{\boldsymbol{\theta}}$ is employed to produce the intervals $\mathbf{z}^I$, which are stored in the list $\mathcal{L}_{\mathbf{z}}^I$. Equi-distribution of the vertices is performed in the attempt to generate well-distributed intervals having similar volume. The algorithm is sketched in the Appendix.

B. Optimizing $\mathbf{P}$ and c for the candidate set

Given a set $S_{P,c}$, component-wise bounds on the state $\boldsymbol{\chi}$ can be obtained considering the extremes of the ellipsoid defined by $V(\boldsymbol{\chi}) = c$. It is evident that different $\mathbf{P}$ matrices and level set c will lead to different bounds for each component. Since finding the optimal $\mathbf{P}$ matrix and level set for the best bound on a specific state variable can be very difficult, a suitable matrix is calculated through numerical optimization, while the level set is found with the aid of a heuristic approach. Before introducing the optimization problem, the heuristic procedure utilized to retrieve an approximate level set is presented.

1) Finding an approximate level set c^*: We would like to find a set $\mathcal{S}_{P,c}$ which envelopes system trajectories as tightly as possible. Given a matrix $\mathbf{P}$, the level value c scales the bounding ellipsoid in state-space. In order to find the optimal c one would need to solve the constrained nonlinear

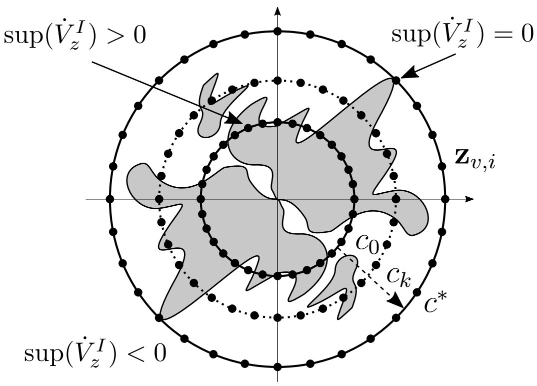


Fig. 2. Visualization of the heuristic procedure presented in section III-B.1. The regions of state-space depicted in gray define areas where $\dot{V}_z$ can assume positive values. At every k-th iteration, the algorithm evaluates $\dot{V}_z^I$ for every vertex $\mathbf{z}_{v,i}$ on the boundary of the current level set $V = c_k$. The maximum attainable value of $\dot{V}_z$ (after iterating over all vertices) is utilized to decide whether to further enlarge c, or to stop the algorithm. For visualization purposes, the vertices are enlarged as the level set grows. Actually, in $\mathbf{z}$ coordinates, the level set of V_z remains 1, while the uncertain regions are deformed by $\mathbf{P}$ and c.

minimization problem

$$\begin{aligned} &\min_c c \\ &\text{subject to } \dot{V}_z(\mathbf{z},t) \le 0, \forall \mathbf{z} \in \|\mathbf{z}\| = 1, \\ &\qquad c > 0, \\ &\qquad \mathbf{w}(t) \in \Omega_\mathbf{w}. \end{aligned} \tag{11}$$

The global minimum of (11) is generally complex to obtain, due to the nonlinearity of the constraints. Since we do not intend to calculate the optimal bound, but rather a sufficiently good one, we aim to find an approximate solution of (11) by means of an heuristics procedure. To this end, the vertices $\mathbf{z}_v$ obtained from algorithm III-A are utilized to evaluate $\dot{V}_z$ on the boundary $\partial\mathcal{S}_{P,c}$. Starting from a value $c = c_0$, the scalar c^* is found by enlarging c with the law

$$c_{k+1} = c_k + \alpha \left| \max_{\mathbf{z}_v \in \mathcal{L}_\mathbf{z}^v} \sup\left(\dot{V}_z^I(\mathbf{z}_v, \mathbf{w}^I)\right)\right|, \quad \alpha \in \mathbb{R}^+, \tag{12}$$

until the derivative $\dot{V}_z(\mathbf{z}_v)$ is strictly negative for all vertices, i.e.,

$$\max_{\mathbf{z}_v \in \mathcal{L}_\mathbf{z}^v} \sup\left(\dot{V}_z^I(\mathbf{z}_v, \mathbf{w}^I)\right) < -\epsilon, \tag{13}$$

where $\sup(\cdot)$ is the supremum operator on an interval, and ϵ is a small positive scalar. The algorithm is sketched in Figure 2. Reasonable values for c_0 could be estimated by performing a batch of simulations and recover the highest value of the level set for the specific $\mathbf{P}$ along the computed trajectories (which can be stored for the next iterations). The retrieved value represents a certain lower bound for the sought level set. Notice that the simulations are not required for the algorithm to work.

2) Optimization problem: Nonlinear constrained optimization (using gradient descent) is hereby used to find a $\mathbf{P}$ matrix which minimizes the bound on a specific component of $\boldsymbol{\chi} = [\boldsymbol{\Phi}^T, \boldsymbol{\Psi}^T]^T$. The reason why bounds on $\boldsymbol{\chi}$ are minimized is that, by inverting the coordinate transformations introduced in (3), a bound on $\boldsymbol{\Phi}_i$ can be directly translated into a bound on a $\mathbf{q}_i$, while bounds on the components of $\boldsymbol{\Psi}$ can be translated into bounds on the components of $\dot{\mathbf{q}}$. The objective function to be minimized, denoted by FINDAPPROXBOUND$(\bar{\mathbf{P}}, i)$, calculates an approximate bound on the i-th component of $\boldsymbol{\chi}$ for a given matrix $\bar{\mathbf{P}}$, by performing the following steps:

1. find an approximate level set c^* using the heuristic described in the previous paragraph III-B.1 and the current guess $\bar{\mathbf{P}}$,
2. calculate and return the bound on the component $\boldsymbol{\chi}_i$ considering the ellipsoid $\boldsymbol{\chi}^T\bar{\mathbf{P}}\boldsymbol{\chi} = c^*$,

Since $\mathbf{P} \in \mathbb{R}^{2n\times 2n}$ is symmetric and scaling is irrelevant, the matrix can be represented with a vector $\mathbf{p} \in \mathbb{R}^{(2n-1)(n+1)}$. The optimization problem can be stated as follow:

$$\begin{aligned} &\min_\mathbf{P} \text{FINDAPPROXBOUND}(\mathbf{P}, i) \\ &\text{subject to } \mathbf{P} \succ 0. \end{aligned} \tag{14}$$

A numerical optimizer is used to tune the coefficients of $\mathbf{P}$ and minimize the bound computed by the objective function.

3) Remarks on the optimization of $\mathbf{P}$*:* The matrix $\mathbf{P}$ is optimized using information on the sign of $\dot{V}_z$ along the boundary, which is calculated solely at the generated vertices. It must be noted that, as such, the function FINDAPPROXBOUND$(\mathbf{P}, i)$ does not return a formal bound, but only a candidate one, since at this phase invariance of the level set is not tested yet. The reason behind this choice is that determining positive invariance by means of IA is a computationally expensive operation, and this process would need to be iterated multiple times to generate the gradient used by a nonlinear optimization algorithm. Fig. 3 depicts some stages during the optimization (14). Reasonable values for the initial value of $\mathbf{P}$ can be calculated by solving the Lyapunov equation for the system at the operating point:

$$\mathbf{A}(\mathbf{0})^T\mathbf{P} + \mathbf{P}\mathbf{A}(\mathbf{0}) = -\mathbf{Q}, \tag{15}$$

where $\mathbf{Q}$ is a positive-definite matrix.

After the optimized matrix $\mathbf{P}^*$ and level set c^* are obtained, a further algorithm is required to prove positive invariance of the set S_{P^*,c^*}, allowing to retrieve guaranteed upper bounds on $\boldsymbol{\chi}$. This algorithm is presented in the following paragraph.

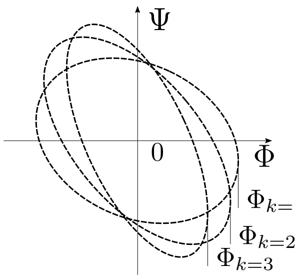
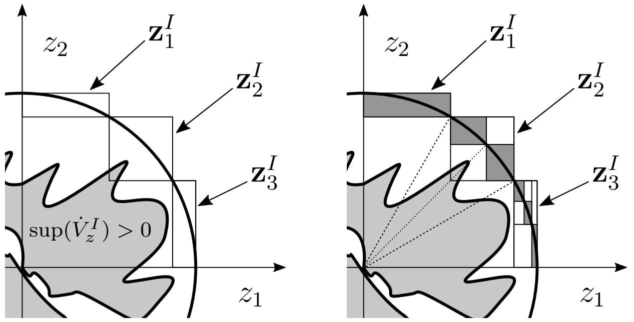


Fig. 3. Visualization of the optimization presented in section III-B. The coefficients of $\mathbf{P}$ are optimized to minimize the bound for a specific component of $\boldsymbol{\chi}$. In this case, the bound on Φ is being minimized. Using eq. (3), this bound will directly translate into a bound for $\tilde{\mathbf{q}}$.



(a) analysis (b) splitting

Fig. 4. Visualization of Algorithm 2, presented in section III-C. The procedure verifies that $\dot{V} \leq 0$ for all points of the boundary $\partial\mathcal{S}_{P^*,c^*}$. Intervals $\mathbf{z}_2^I$ and $\mathbf{z}_3^I$ require splitting. The angles generating the intervals are bisected and tighter interval enveloping the boundary are generated. The interval boxes fulfilling the negativity condition are colored in dark gray. The interval $\mathbf{z}_3^I$ requires to apply the splitting procedure multiple times.

C. Prove positive invariance via boundary analysis

The set $\mathcal{S}_{P^*,c^*}$ is positive invariant if the following condition is satisfied:

$$\forall \mathbf{z} \in \|\mathbf{z}\| = 1 : \dot{V}_z(\mathbf{z}, t) \leq 0. \tag{16}$$

Before checking condition (16), the level set c^* is increased to account for the box approximation. Being $\mathbf{z}_{s,n}$ the closest vertex to the origin, then the new level set $\hat{c}$ is

$$\hat{c} = c^* + \Delta c^*, \quad \Delta c^* = c^*(1 - \mathbf{z}_{s,n}^T \mathbf{z}_{s,n}). \tag{17}$$

The enlarged set $\mathcal{S}_{P^*,\hat{c}}$ is utilized henceforth.

Each multidimensional interval $\mathbf{z}_i^I$ stored in the list $\mathcal{L}_{\mathbf{z}}^I$ is evaluated to satisfy the condition

$$\sup\left(\dot{V}_z^I(\mathbf{z}_i^I, \mathbf{w}^I)\right) \leq 0. \tag{18}$$

If all intervals satisfy this condition, system trajectories in $\mathbf{z}$ coordinates cannot escape the unit-hypersphere. Even if $\dot{V}_z$ is always negative inside of a considered interval, (18) might be positive due to the conservative nature of IA. If condition (18) is not met, the polar angles $\boldsymbol{\theta}_i$ associated with the interval $\mathbf{z}_i^I$ are bisected along each dimensions and used to generated new intervals, which are subsequently tested. Splitting reduces the inherent conservativeness of IA [22]. This is performed by the functions $\textsc{Split}(\mathbf{z}_i^I, \boldsymbol{\theta}_i^I)$ and $\textsc{GetPolarSplit}(\mathbf{z}_i^I, \boldsymbol{\theta}_i^I)$. The pseudo-code in Algorithm 2 summarizes the aforementioned steps. Using different parameterizations to describe subsets can further reduce conservativeness: the effects of dependency can be reduced by using generalized IA, affine formulations, and higher order methods [18], but we do not dwell further on this topic in this article.

If Algorithm 2 succeeds to prove that $\dot{V}_z < 0$ along the entire boundary, the set $\mathcal{S}_{P,\hat{c}}$ can be finally used to retrieve bounds on the energy coordinates and on the original states. Instead, if the algorithm fails the proof, it is necessary to increase $\hat{c}$, and check the boundary again. Alternatively, the whole procedure can be performed again with an increased number of vertices in Algorithm 1.

Algorithm 2 Boundary Analysis

```
procedure BOUNDARY_ANALYSIS($\mathbf{P}, c, \mathcal{L}_{\mathbf{z}}^I, \mathcal{L}_{\theta}^I$)
    isValid = true;
    for $\mathbf{z}_i^I \in \mathcal{L}_{\mathbf{z}}^I$ do
        if $\sup(\dot{V}_z^I(\mathbf{z}_i^I)) > 0$ then
            isNegative = SPLIT($\mathbf{z}_i^I, \boldsymbol{\theta}_i^I$);
            if not isNegative then
                isValid = false;
                break;
    return isValid;
end procedure

function SPLIT($\mathbf{z}_i^I, \boldsymbol{\theta}_i^I$)
    if $\max(\mathrm{rad}(\mathbf{z}_i^I)) < \varepsilon$ then
        return false;
    [$\mathbf{z}_1^I, \boldsymbol{\theta}_1^I, \mathbf{z}_2^I, \boldsymbol{\theta}_2^I$] = GETPOLARSPLIT($\mathbf{z}_i^I, \boldsymbol{\theta}_i^I$);
    if $\sup(\dot{V}_z^I(\mathbf{z}_1^I)) < 0$ and $\sup(\dot{V}_z^I(\mathbf{z}_2^I)) < 0$ then
        return true;
    else
        if $\sup(\dot{V}_z^I(\mathbf{z}_1^I)) < 0$ then
            return SPLIT($\mathbf{z}_2^I, \boldsymbol{\theta}_2^I$);
        else
            if $\sup(\dot{V}_z^I(\mathbf{z}_2^I)) < 0$ then
                return SPLIT($\mathbf{z}_1^I, \boldsymbol{\theta}_1^I$);
            else
                return SPLIT($\mathbf{z}_1^I, \boldsymbol{\theta}_1^I$) and SPLIT($\mathbf{z}_2^I, \boldsymbol{\theta}_2^I$);
end function
```

D. Retrieve bounds for the original system

The extreme values of the components of $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ are successively converted to the intervals $\boldsymbol{\Phi}^I$ and $\boldsymbol{\Psi}^I$, respectively. Given $\boldsymbol{\Phi}^I$, component-wise bounds on $\tilde{\mathbf{q}}$ are given by

$$\tilde{\mathbf{q}}_b^I = \sqrt{2} \cdot \mathbf{K}^{-\frac{1}{2}} \cdot \boldsymbol{\Phi}^I. \tag{19}$$

Given a bound on $\boldsymbol{\Psi}$, a bound on $\dot{\mathbf{q}}$ can be calculated as

$$\|\dot{\mathbf{q}}_b\| \leq \sqrt{2} \cdot \max_{\forall \mathbf{q} \in \mathcal{C}^n} \left\|\mathbf{M}(\mathbf{q})^{-\frac{1}{2}}\right\| \|\boldsymbol{\Psi}\|, \tag{20}$$

or, using IA, component-wise bounds for $\dot{\mathbf{q}}$ can be calculated as

$$\dot{\mathbf{q}}_b^I = \sqrt{2} \cdot \mathbf{M}(\mathbf{q}_b^I)^{-\frac{1}{2}} \cdot \boldsymbol{\Psi}^I, \tag{21}$$

where $\mathbf{q}_b^I$ can be extracted from $\tilde{\mathbf{q}}_b^I$.

IV. Examples

We present an example of application of the proposed algorithm to a 1-DOF non-linear mass-spring-damper system affected by disturbance. We then proceed to apply the method to a 2-DOF robot arm. The results are compared to randomized simulations and ultimate boundedness. For the examples in this work, the MATLAB implementation of interval arithmetic of the CORA Toolbox [24] has been utilized..

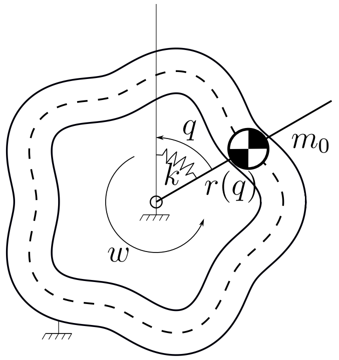


Fig. 5. Sketch of system (22). The sliding guide depicted with the dashed profile constraints the motion of the mass m_0 in function of the angle q.

A. 1-DOF nonlinear system

Let us consider the system depicted in Figure 5. This 1-DOF nonlinear system is described by the equation

$$m(q)\ddot{q} + c(q,\dot{q})\dot{q} + k\tilde{q} + d\dot{q} = w(t), \tag{22}$$

where

$$\begin{aligned} m(q) &= m_0 r(q)^2, \quad r(q) = r_0 + r_\Delta \cos(hq), \\ c(q,\dot{q}) &= -h m_0 r_\Delta \sin(hq) r(q)\dot{q}, \end{aligned} \tag{23}$$

The point mass $m_0 = 1$ [kg] is constrained to the path described by $r(q)$. The parameters $r_0 = 0.1$ [m], $r_\Delta = 0.01$ [m], $h = 5$ are used to generate the path depicted in Figure 5. The amplitude of the disturbance is bounded: $\|w(t)\| \leq 1$. The gains k and d are designed to achieve a desired damping ratio ζ at the operation point $q_d = 0$. For the first test, we consider $k = 1$ [N/m] and $d = 0.22$ [Ns/m], corresponding to $\zeta = 1$. The whole algorithm presented in Section III is applied to find a positive invariant region in the neighborhood of the origin. In the following, each step of the algorithm is described in detail. First, 500 vertices are generated on the unitary hypersphere using Algorithm 1. Then, system (22) is transformed to energy coordinates $\boldsymbol{\chi} = [\Phi, \Psi]^T \in \mathbb{R}^{2\times 1}$ according to (3):

$$\begin{aligned} \dot{\Phi} &= +\sqrt{\frac{k}{m(\Phi)}}\Psi \\ \dot{\Psi} &= -\sqrt{\frac{k}{m(\Phi)}}\Phi - \frac{d}{m(\Phi)}\Psi + \frac{1}{\sqrt{2m(\Phi)}}w(t), \end{aligned} \tag{24}$$

where the substitutions $q = \sqrt{\frac{2}{k}}\Phi$ and $\dot{q} = \sqrt{\frac{2}{m(\Phi)}}\Psi$ have been performed. The skew-symmetric term $\mathbf{S}$, previously introduced in (4), does not appear in the dynamics of system (24). Therefore, in the 1-DOF case, one gets rid of the gyroscopic terms thanks to the "energy" coordinate transformation. The optimization of $\mathbf{P}$ (step III-B) is performed to minimize the bound on Φ (hence on q). The optimization outputs the matrix

$$\mathbf{P}_1^* = \begin{bmatrix} 0.6167 & 0.2285 \\ 0.2285 & 1.0000 \end{bmatrix}, \tag{25}$$

and a level set $c_1^* = 3.264 \cdot 10^{-1}$. Subsequently, we generate 1000 vertices on the hypersphere and envelope this boundary with interval boxes using Algorithm 1 (to refine the border approximation). The level set is enlarged according to (17) to obtain $\hat{c}_1 = 3.285 \cdot 10^{-1}$. Afterwards, the boundary analysis algorithm (section III-C) is employed and the set $\mathcal{S}_{P_1^*, \hat{c}_1}$ is proven to be positive invariant. The validated bounds on Φ and Ψ are found to be:

$$\Phi \in [-0.7628, 0.7628], \quad \Psi \in [-0.5990, 0.5990]. \tag{26}$$

TABLE I

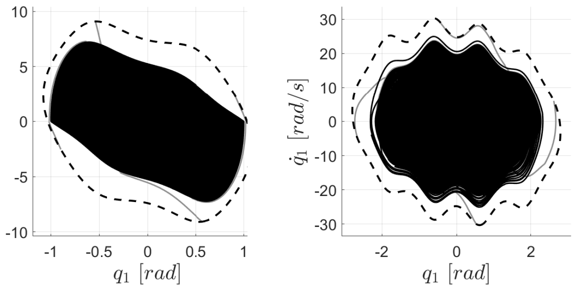
BOUNDS ON THE STATES OF THE 1-DOF SYSTEM.

Disturbance [Nm]	Method	$\lvert\tilde{q}\rvert$ [rad]	$\lvert\dot{q}\rvert$ $\left[\frac{\text{rad}}{\text{s}}\right]$
Test 1 $\mathbf{w} \in [-1, 1]$	Randomized Simulation	$\geq$1.000	$\geq$7.218
	Proposed Method	$\leq$1.079	$\leq$9.413
	Ultimate Boundedness	$\leq$21.934	$\leq$252.324
Test 2 $\mathbf{w} \in [-1, 1]$	Randomized Simulation	$\geq$2.306	$\geq$24.385
	Proposed Method	$\leq$2.791	$\leq$30.523
	Ultimate Boundedness	$\leq$84.055	$\leq$942.224

Finally, the bounds on q and $\dot{q}$ are calculated using (19) and (21). Table I presents a comparison of the bounds on q and $\dot{q}$ determined by the proposed method and by ultimate boundedness applied in configuration space for this first test. To find ultimate bounds, the method described in [12] is applied on system (22), considering zero desired trajectory tracking speed and acceleration. In order to highlight the validity and usefulness of the proposed method, we have also performed randomized simulations and included the results in Table I. The disadvantage of simulations is that they can only prove that a set can be escaped, but they cannot prove that a set cannot be escaped [19, Sec. 1]. Since the system is continuous, an infinite number of simulations would be required to check all possible realization of the disturbance and initial conditions. Since simulations cannot provide an upper bound on the states, a direct comparison with the proposed method is not possible. This relevant fact is made explicit by means of inequality signs in the entries of Table I. Nevertheless, the real bounds must lie between these two results, thus the closer these bounds are to each other, the tighter the estimated bounds are to the real ones. For every test, 10^2 simulations are carried out. For these simulations, the disturbance is realized by randomly switching its extreme values. The number of switches per simulation is also randomized. The randomized simulations and the invariant region found by the proposed method are depicted in Fig. 6a. Table I also presents the results obtained in a second test, by considering $k = 1$ [N/m] and $d = 0.055$ [Ns/m], corresponding to a lower damping ratio $\zeta = 0.25$. The simulation trajectories and the invariant region are depicted in Fig. 6b.

B. 2-DOF nonlinear system

Let us now consider a 2-DOF robotic arm depicted in Figure 7. The kinematic and dynamic parameters of this system are presented in Table II. The initial configuration of the robot ($\mathbf{q}_0 = [-\frac{\pi}{4}, -\frac{\pi}{2}]^T$) is utilized as operating point.



(a) Test 1 (b) Test 2

Fig. 6. The figure depicts simulated trajectories (solid lines), and the border of the invariant region found by the proposed method (thick dashed line) in the two tests. The real bounds must lie between these two results, thus the closer these bounds are to each other, the tighter the estimated bounds are to the real ones. For both tests, four additional simulated trajectories with initial condition on the border are depicted in gray. The bounds provided by ultimate boundedness are not depicted because they are several magnitudes bigger and the invariant regions could not be visualized.

TABLE II
SETS OF PARAMETERS FOR THE 2-DOF PLANAR ROBOT.

m_1 [kg]	m_2 [kg]	l_1 [m]	l_2 [m]
5	5	1	1

The proportional and derivative gains are designed as

$$\mathbf{K} = \begin{bmatrix} 500 & 0 \\ 0 & 500 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 168.9 & 38.3 \\ 38.3 & 92.4 \end{bmatrix},$$

such that the system is critically damped at the operating point. For this system, we assume a disturbance with different maximum amplitudes for each component. The whole procedure is performed twice in order to find a sharp bound for each of the two position variables. Table III presents a comparison of the bounds on the joints deviations $\tilde{\mathbf{q}}$ obtained by using the different methods.

V. Discussion

The proposed procedure allows to obtain reasonable tight upper bounds on the deviations, which can be used for controller tuning, and the evaluation of worst-case performance. This result is achieved thanks to a good matching between the morphology of the real invariant state-space region and the one of the optimized invariant set produced by the method. The particular morphology is obtained implicitly using a coordinate transformation, which essentially performs a nonlinear scaling.

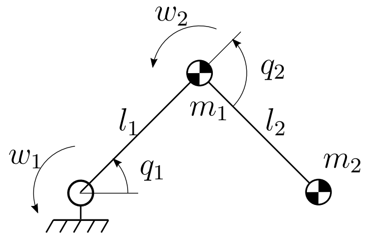


Fig. 7. Visualization of the 2-DOF serial robot.

TABLE III
BOUNDS ON THE STATES OF THE 2-DOF SYSTEM.

Disturbance [Nm]	Method	$\lvert\tilde{q}_1\rvert$ [rad]	$\lvert\tilde{q}_2\rvert$ [rad]
$\mathbf{w} \in \begin{bmatrix} [-1,1] \\ [-10,10] \end{bmatrix}$	Randomized Simulation	$\geq$0.010	$\geq$0.021
	Proposed Method	$\leq$0.019	$\leq$0.040
	Ultimate Boundedness	$\leq$0.793	$\leq$0.793

TABLE IV
MAX COMPUTATIONAL TIMES

System	1-DOF	2-DOF
Optimization of $\mathbf{P}$ [sec]	15	80
Boundary Analysis [sec]	5	370
Total Time [sec]	20	450

Sources of conservativeness: It is difficult to evaluate how close the results are to the exact worst-case, since choosing the ”worst-case disturbance” for the randomized simulations (such that the bounds are not too optimistic) is not possible in general. Despite providing reasonably good results, the proposed method retains a certain level on conservativeness due to the choice of using ellipsoids through (5). The bounds could be tightened by not restricting ourselves to estimates in ellipsoidal form [8, Chapter 8.2], for example by constructing polyhedral invariant sets [21] starting from the results of the method proposed in this paper. However, a drawback of polyhedral sets is that complexity increases further.

Optimization: Even if the search for an optimal matrix $\mathbf{P}$ involves performing non-linear optimization for each state of interest, ultimately it may be sufficient to run the algorithm only once and find “good enough” bounds on all the states. In this case, the optimization of $\mathbf{P}$ can be carried out considering $\|\mathbf{\Phi}\|$ as objective function to be minimized in (14).

Features: The method does not require discretization of the dynamics (used for example in [20]), neither relies on linearization around the operating point. Additionally, an explicit symbolic formulation of the dynamics is not necessarily needed: it is only required to compute the interval matrices and vectors for given ranges. This can be achieved by extending an existing dynamic library to allow interval computations.

Complexity: The number of free parameters to be optimized in (14) increases exponentially with the number of DOF. The number of vertices (and interval boxes) required to discretize the boundary with a given resolution increases also exponentially. Table IV presents the computational times required to obtain the results in Section IV. The tests were run on a laptop equipped with a 1.90GHz quad-core processor. The computational times grow quickly with the system dimension. The trade-off between complexity and conservativeness represents one of the main drawbacks of branch and bound numerical approaches applied in the state-space. It constitutes one of the biggest issue left to be solved in this field of research. Nevertheless, there is room for improvement in efficiency, by switching to better set representations for the boundary. Moreover, the heaviest computational parts are completely parallelizable, hence, reasonable computational times could be obtained by switching to an appropriate architecture.

VI. Conclusion

We presented a numerical algorithm based on invariant set theory to estimate provable positive invariant regions for nonlinear mechanical systems affected by disturbances. The invariant sets are used to infer worst-case deviations of system trajectories from an operating point. A coordinate transformation was carried out in order to simplify the computation of invariant regions in state-space. Our results were compared to ultimate boundedness via numerical simulation. The simulations showed that the proposed method proved to yield tighter component-wise bounds on the states, which are useful for the analysis of worst-case performance and tuning of the controller.

Future research will primarily focus on extending the method to handle a varying equilibrium point and the tracking case, also considering different types of perturbations, like state-dependent perturbations and model mismatch. Furthermore, we plan to improve the efficiency throughout a better set representation, and to assess the computational feasibility for a high number of DOFs.

Appendix I
Equi-distributed vertices and intervals on the n-sphere

Algorithm 1 Equi-distributed vertices on the n-sphere

```
procedure DISTRIBUTE_VERTICES($n, r, N$)
    $a = $ HYPERSPHERE_SURFACE$(n, r)/N$;
    for $i = 1 \ldots n-2$ do
        $d(i) = \sqrt[n-i]{d_{prod}}$;
        $M_\theta(i) = \text{round}(\pi/d(i))$;
        $d_\theta(i) = \pi/M_\theta(i)$;
        $d_{prod} = d_{prod}/d_t(i)$;
    $d_\theta(n-1) = d_{prod}$;
    $N_{count} = 0$;
    PREALLOCATE($\boldsymbol{\theta}, N$);
    GENERATEANGLES($1, M_\theta(1)$);
    $\mathbf{z}_v = $ GENERATEVERTICES($\boldsymbol{\theta}$);
end procedure

function GENERATEANGLES($i, M$)
    $r = \pi$;
    if $i == n-2$ then
        $r = 2\pi$;
    for $j = 1 \ldots M(i)$ do
        if $i < n-1$ then
            $\theta(i) = \pi\left((j-1)+0.5\right)/M$;
            $M_{next} = \text{round}\left(\frac{r}{d_\theta(i+1)}\prod_1^i \sin(\theta(i))\right)$;
            GENERATEANGLES($i+1, M_{next}$);
        else
            $\theta(i) = 2\pi\left(j-1\right)/M$;
            $N_{count} = N_{count} + 1$;
            $\boldsymbol{\theta}(N_{count}) = [\theta(1), \theta(2), \ldots, \theta(n-1)]$;
end function
```

References

[1] M. Green and D. J. Limebeer, *Linear robust control*. Dover Publications, 2012.

[2] R. Konwar, "A review on sliding mode control: An approach for robust control process," *ADBU Journal of Electrical and Electronics Engineering (AJEEE)*, vol. 1, no. 1, pp. 6–13, 2017.

[3] D. Q. Mayne, E. C. Kerrigan, E. J. van Wyk, and P. Falugi, "Tube-based robust nonlinear model predictive control," *Intern. Journal of Robust and Nonlinear Control*, vol. 21, no. 11, pp. 1341–1353, 2011.

[4] F. Allgöwer and A. Zheng, *Nonlinear model predictive control*. Birkhäuser, 2012, vol. 26.

[5] W. Yu, X. Li, and R. Carmona, "A novel PID tuning method for robot control," *Industrial Robot: An International Journal*, vol. 40, no. 6, pp. 574–582, Oct. 2013.

[6] Kiam Heong Ang, G. Chong, and Yun Li, "Pid control system analysis, design, and technology," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 4, pp. 559–576, July 2005.

[7] C. Ott, A. Albu-Schaffer, A. Kugi, and G. Hirzinger, "On the passivity-based impedance control of flexible joint robots," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 416–429, April 2008.

[8] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.

[9] S. Kawamura, F. Miyazaki, and S. Arimoto, "Is a local linear pd feedback control law effective for trajectory tracking of robot motion?" in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*. IEEE, 1988, pp. 1335–1340.

[10] J. Wang, S. J. Dodds, and W. N. Bailey, "Guaranteed rates of convergence of a class of PD controllers for trajectory tracking problems of robotic manipulators with dynamic uncertainties," *IEE Proc. - Control Theory and Applications*, vol. 143, no. 2, pp. 186–190, Mar. 1996.

[11] 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.

[12] Q. Chen, H. Chen, Y. Wang, and P.-Y. Woo, "Global stability analysis for some trajectory-tracking control schemes of robotic manipulators," *Journal of Robotic Systems*, vol. 18, no. 2, pp. 69–75, 2001.

[13] J. H. Kim, S.-M. Hur, and Y. Oh, "Performance analysis for bounded persistent disturbances in pd/pid-controlled robotic systems with its experimental demonstrations," *International Journal of Control*, vol. 91, no. 3, pp. 688–705, 2018.

[14] E. Kofman, H. Haimovich, and M. M. Seron, "A systematic method to obtain ultimate bounds for perturbed systems," *International Journal of Control*, vol. 80, no. 2, pp. 167–178, 2007.

[15] H. Haimovich and J. L. Mancilla-Aguilar, "On bounding a nonlinear system with a monotone positive system," in *2017 XVII Workshop on Information Processing and Control (RPIC)*, Sep. 2017, pp. 1–6.

[16] G. Bitsoris, M. Vassilaki, and N. Athanasopoulos, "Robust positive invariance and ultimate boundedness of nonlinear systems," in *20th Mediterranean Conf. on Control Automation*, Jul. 2012, pp. 598–603.

[17] H. Haimovich, E. Kofman, and M. M. Seron, "Analysis and Improvements of a Systematic Componentwise Ultimate-bound Computation Method," *IFAC Proc. Vol.*, vol. 41, no. 2, pp. 1319–1324, Jan. 2008.

[18] B. Chachuat, B. Houska, R. Paulen, N. Peri'c, J. Rajyaguru, and M. E. Villanueva, "Set-theoretic approaches in analysis, estimation and control of nonlinear systems," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 981–995, 2015.

[19] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *2008 47th IEEE Conference on Decision and Control*, Dec. 2008, pp. 4042–4048.

[20] J. M. Bravo, D. Limón, T. Alamo, and E. F. Camacho, "On the computation of invariant sets for constrained nonlinear systems: An interval arithmetic approach," *Automatica*, vol. 41, no. 9, pp. 1583–1589, 2005.

[21] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*. Springer International Publishing, 2015.

[22] T. Hickey, Q. Ju, and M. H. Van Emden, "Interval arithmetic: From principles to implementation," *Journal of the ACM (JACM)*, vol. 48, no. 5, pp. 1038–1068, 2001.

[23] M. Deserno. (2004) How to generate equidistributed points on the surface of a sphere. [Accessed: 10-July-2019]. [Online]. Available: https://www.cmu.edu/biolphys/deserno/pdf/sphere_equi.pdf

[24] M. Althoff and D. Grebenyuk, "Implementation of interval arithmetic in CORA 2016," in *Proc. of the 3rd International Workshop on Applied Verification for Continuous and Hybrid Systems*, 2016, pp. 91–105.