Safety Index Synthesis via Sum-of-Squares Programming

Weiye Zhao¹, Tairan He¹, Tianhao Wei, Simin Liu and Changliu Liu

Abstract—Control systems often need to satisfy strict safety requirements. Safety index provides a handy way to evaluate the safety level of the system and derive the resulting safe control policies. However, designing safety index functions under control limits is difficult and requires a great amount of expert knowledge. This paper proposes a framework for synthesizing the safety index for general control systems using sum-of-squares programming. Our approach is to show that ensuring the non-emptiness of safe control on the safe set boundary is equivalent to a local manifold positiveness problem. We then prove that this problem is equivalent to sum-of-squares programming via the Positivstellensatz of algebraic geometry. We validate the proposed method on robot arms with different degrees of freedom and ground vehicles. The results show that the synthesized safety index guarantees safety and our method is effective even in high-dimensional robot systems.

I. INTRODUCTION

Energy-function-based algorithms [1] have been widely studied as appealing tools for safe control. When properly designed, the energy function (also called the safety index or barrier function) maps dangerous states to high values and safe states to low values. Then, safety can be ensured by finding a control input that dissipates the energy. However, when control limits exist, this desired safe control may not be realizable and thus the safety guarantee may be broken. Therefore, it is important to account for control limits in energy function design.

Accounting for control limits in safety index design is challenging, because we have to ensure the feasibility of safe control at infinitely many states. This feasibility constraint itself can be quite complex, as it is a function of the control limits, system dynamics, and safety requirements. There have been various approaches to safety index synthesis under control limits. Some works assume that the dynamical system has a special structure (i.e. kinematic bicycle, Euler-Lagrange), which enables hand-derivation of a safety index [2], [3]. Another approach is to formulate synthesis as optimizing parameters within some safety index function form [4], [5]. However, these methods do not scale well to high-dimensional systems.

Sum-of-Squares Programming (SOSP) has been applied to safety index synthesis [4], [6], as it is a powerful tool for dealing with optimization problems with infinitely many constraints. The issue with existing SOSP-based synthesis techniques is that they try to enforce existence of safe control for all states in the state space [4], [6]. This is otherwise known as enforcing a global positiveness constraint [7].

In this work, we highlight that instead of satisfying global positiveness constraints, it is sufficient for the safety index to satisfy local manifold positiveness constraints, which enables the corresponding safety index design to be much less conservative. That means we can find solutions (a valid safety index) more often. Subsequently, we propose a general, scalable, efficient SOSP-based method to transform safety index synthesis problem into a nonlinear programming problem that can be solved efficiently by off-the-shelf solvers.

The experiments show that the synthesized safety index ensures the existence of safe control and that our method is time-efficient, even for high-dimensional robotic systems.

Our contributions We propose a novel SOSP-based method that can efficiently synthesize a safety index. The state-of-the-art SOSP-based methods are (i) limited to systems with polynomial or sinusoidal nonlinearities [6]; (ii) hard to scale to high-dimensional systems (e.g., iteratively solving $O(2k)$ SOS programs at $k$-th iteration [4]); (iii) limited to conservative solutions due to considering global positiveness constraints. Our algorithm (i) is applicable to polynomial substitutable (Def. 1) control systems; (ii) scales well to high-dimensional systems (solving only one nonlinear programming) and (iii) gives nonconservative solutions via solving local manifold positiveness constraints.

In the remainder of the paper, we first discuss related work about energy-function-based methods and safe control in Section II. We then formulate the mathematical problem for safety index synthesis in Section III. In Section IV and Section V we first introduce the proposed optimization algorithm based on SOSP, and provide theoretical results that the algorithm can obtain a feasible safety index design. Finally, we validate our proposed method in high-dimensional robot systems in Section VII. Our code is available on Github.

II. RELATED WORK

In literature, many different energy functions [1] are proposed to measure safety, including (i) potential function; (ii) safety index; (iii) control barrier function. Representative methods include potential field method, barrier function method [8], and safe set algorithm [9].

However, designing such energy functions is difficult and requires great human efforts to find appropriate parameters.
and function forms. As for synthesizing safety index, [9] discusses the general safety index design rule that guarantees forward invariance of safety. However, the safety guarantee comes from the assumption of unbounded control input. [2] solves the problem based on worst-case analysis, but it is limited to simple 2D mobile robot dynamics. [5] leverages an evolutionary algorithm for safety index parameterization, but the computation time of grid sampling increases exponentially with the dimension of the state space. [10] proposes to learn approximate CBFs with neural networks, but these methods lack theoretical guarantee. Some automated methods [4] develop a SOSP-based program via Positivstellensatz to verify CBFs, but it is also computationally challenging for high-dimensional systems. [11] proposes synthesis of certificate functions in complex systems, another line of work ([8], [12]) proposes to learn approximate CBFs to designs certificate functions. However, [4] is limited to simple 2D mobile robot dynamics. [5] leverages SOSP to synthesize CBFs. However, [4] is limited to simple 2D mobile robot dynamics. [8] proposes synthesis of CBFs via iterative optimization to eliminate inevitably-unsafe states. Then, safe control can be applied as the negative closest distance between the robot and environmental obstacles.

Problem: Since \( \mathcal{X}_S \) may contain states that will inevitably go to the unsafe set no matter what the control input is, we need to assign high energy values to these inevitably-unsafe states. Then, safe control can be applied to ensure forward invariance in a subset of the safe set \( \mathcal{X}_S \) by dissipating the energy. Forward invariance of a set means that the robot state will never leave the set if it starts from the set, i.e., when \( \phi_0(x(t_0)) \leq 0 \), then \( \phi_0(x(t)) \leq 0 \), \( \forall t > t_0 \).

In this paper, we adopt safe set algorithm (SSA) [9], which is an energy function-based method for safe control. SSA has introduced a rule-based approach to synthesize the energy function as a continuous, piece-wise smooth scalar function \( \phi : \mathbb{R}^{n_x} \to \mathbb{R} \). And the energy function \( \phi(x) \) is called a safety index. The general form of the safety index is \( \phi = \phi_0 + k_1\phi_0^{(1)} + \cdots + k_n\phi_0^{(n)} \) where 1) the roots of \( 1 + k_1s + \cdots + k_ns^n = 0 \) are all negative real (to ensure zero-overshooting of the original safety constraints); 2) the relative degree from \( n \)-order derivative \( \phi^{(n)}_0 \) to \( u \) is one (to avoid singularity). In our paper, we assume \( \frac{\partial \phi_0}{\partial x} \) satisfies Definition 1.

It is shown in [9] that if the control input is unbounded (\( \mathcal{U} = \mathbb{R}^{n_u} \)), then there always exist a control \( u \) that satisfies the constraint \( \phi(x, u) \leq 0 \) when \( \phi = 0 \), where \( \phi(x, u) \) denotes the derivative of \( \phi(x) \) with respect to time under control \( u \). For simplicity, we use \( \dot{\phi} \) to represent \( \phi(x, u) \). If the control input always satisfies that constraint, then the set \( \{ x \mid \phi(x) \leq 0 \} \) is forward invariant. In practice, when \( \phi = 0 \), the safe control \( u \) is computed through a quadratic projection of the nominal control \( u^* \):

\[
\begin{align*}
\dot{x} &= f(x) + g(x)u, \\
\text{where } f : \mathbb{R}^{n_x} &\to \mathbb{R}^{n_x}, \quad g : \mathbb{R}^{n_x} &\to \mathbb{R}^{n_x \times n_u} \text{ are locally Lipschitz continuous on } \mathbb{R}^{n_x}. \text{ Here we highlight that } f(x), g(x) \text{ are not necessarily polynomials, and they satisfy the following definition:}
\end{align*}
\]

**Definition 1:** (Substitutable Function) A function is substitutable if its variables can be projected into the higher dimensional polynomial manifold, such that the function can be rewritten as polynomials.

**Remark 1:** To better understand Definition 1, we give an example of a substitutable function. Consider function

\[
f^*(\theta) := \sin \theta \cos \theta,
\]

by substituting \( \sin \theta \) with \( a \), and \( \cos \theta \) with \( b \), respectively. \( f^*(\theta) \) can be rewritten as following polynomial

\[
f^*(a, b) := ab,
\]

where \( a^2 + b^2 - 1 = 0 \).

Safety Specification: The safety specification requires the system state should be constrained in a closed subset in the state space, called the safe set \( \mathcal{X}_S \). The safe set can be represented by the zero-sublevel set of a continuous and piecewise smooth function \( \phi_0 : \mathbb{R}^{n_x} \to \mathbb{R} \), i.e., \( \mathcal{X}_S = \{ x \mid \phi_0(x) \leq 0 \} \). \( \mathcal{X}_S \) and \( \phi_0 \) are directly specified by users. The design of \( \phi_0 \) is straightforward in most scenarios. For example, for collision avoidance, \( \phi_0 \) can be designed as the negative closest distance between the robot and environmental obstacles.

**Problem:** Since \( \mathcal{X}_S \) may contain states that will inevitably go to the unsafe set no matter what the control input is, we need to assign high energy values to these inevitably-unsafe states. Then, safe control can be applied to ensure forward invariance in a subset of the safe set \( \mathcal{X}_S \) by dissipating the energy. Forward invariance of a set means that the robot state will never leave the set if it starts from the set, i.e., when \( \phi_0(x(t_0)) \leq 0 \), then \( \phi_0(x(t)) \leq 0 \), \( \forall t > t_0 \).
The issue with (3) is that this safe controller can saturate as shown in Fig. 1 where the sets of safe control may have no intersection with the limited control space. Specifically, saturation occurs when there does not exist a control input that satisfies the constraint $\dot{\phi} \leq 0$ when $\phi = 0$, causing the loss of safety guarantees. This can happen because we have not yet accounted for control limits in our design of $\phi$. Hence, the core problem of this paper is to choose a proper parameterization of safety index that guarantees the existence of safe control (within control limits) to ensure $\dot{\phi} \leq 0$ for all states where $\phi = 0$. We call this process as Safety Index Synthesis with Control Limits. Mathematically, Safety Index Synthesis solves the following problem:

**Problem 1 (Safety Index Synthesis):** Construct safety index as $\phi(x) = \phi_0 + k_1 \phi_1 + \cdots + k_n \phi_n$, with tunable parameters set $\Theta = \{k_1, k_2, \cdots, k_n\}$, and $\forall i = 1, 2, \cdots, n, k_i \in \mathbb{R}^+$, such that

$$\forall x \in \mathcal{X} \text{ s.t. } \phi_0(x) = 0, \min_{u \in U} \hat{\phi}(x, u) < 0. \tag{4}$$

However, solving Problem [1] is not a trivial task. Note that (4) contains infinitely many constraints since for every state $x$, $\text{s.t. } \phi_0(x) = 0$, poses an inequality constraint, hence cannot be directly solved by off-the-shelf nonlinear programming solvers. To deal with this issue, we leverage tools from SOSTOLs (to be reviewed in the following section) to reexpress the problem as a nonlinear program.

**IV. BACKGROUND OF SUM OF SQUARE PROGRAMMING**

Optimization problems with global positiveness constraints [7] in the form of finding a function $F(x)$, s.t. $\forall x \in \mathbb{R}^n, G \cdot F(x) > 0$, where $G$ is a function, have been widely studied and can be solved by SOSTOLs [7]. However, our problem requires $\min_{u \in U} \hat{\phi}(x, u)$ to be nonpositive on a limited-size manifold, i.e. $x \in \mathcal{X}, \phi(x) = 0$. Hence, Problem [1] cannot be directly solved by SOSTOLs. Instead, we will reconsider the theory behind SOSTOLs to solve Problem [1].

To ensure global positiveness of a condition, the easiest way is to show that there does not exist any solution such that the condition is violated (refute set). Constructing the refute set and showing it is empty is the core idea behind SOSTOLs [14] to ensure global positiveness. To show that the refute set is empty, we need to invoke the equivalence conditions in Positivstellensatz [14]. Before introducing Positivstellensatz, we first review a key concept: ring-theoretic cone [14].

**Definition 2 (Ring-theoretic cone):** Denote $\mathbb{R}[x_1, \ldots, x_n]$ a set of polynomials with $[x_1, \ldots, x_n]$ as variables. For a set $S = \gamma_1, \ldots, \gamma_s \subseteq \mathbb{R}[x_1, \ldots, x_n]$, the associated ring-theoretic cone can be expressed as:

$$\Gamma = \{p_0 + p_1 \gamma_1 + \cdots + p_s \gamma_s : p_1 \gamma_1 + \cdots + p_s \gamma_s \gamma_1 + \cdots + p_s \gamma_s \gamma_1 \cdots \gamma_s\},$$

where $p_0, \ldots, p_{12\ldots s}$ are the polynomials that are SOS.

Based on the ring-theoretic cone, Positivstellensatz condition is specified in the following theorem.

**Theorem 1 (Positivstellensatz):** Let $(\gamma_j)_{j=1, \ldots, s}$, $(\psi_k)_{k=1, \ldots, t}$, $(\zeta_l)_{l=1, \ldots, r}$ be finite families of polynomials in $\mathbb{R}[x_1, \ldots, x_n]$. Let $\Gamma$ be the ring-theoretic cone generated by $(\gamma_j)_{j=1, \ldots, s}$, $\Xi$ the multiplicative monoid [14] generated by $(\psi_k)_{k=1, \ldots, t}$, and $\Phi$ the ideal [14] generated by $(\zeta_l)_{l=1, \ldots, r}$. Then, the following properties are equivalent:

1) The following set is empty

$$\{x \in \mathbb{R}^n \mid \gamma_j(x) \geq 0, j = 1, \ldots, s \} \cup \{\psi_k(x) \neq 0, k = 1, \ldots, t \} \cup \{\zeta_l(x) = 0, l = 1, \ldots, r \} = \emptyset. \tag{5}$$

2) There exist $\gamma \in \Gamma$, $\psi \in \Xi$, $\zeta \in \Phi$ such that

$$\gamma + \psi^2 + \zeta = 0, \tag{6}$$

where $\psi = 1$ if $t = 0$.

We refer the reader to [14] for the proofs of Theorem [1]. In summary, Positivstellensatz shows that the refute set being empty is equivalent to a feasibility problem in (6). Therefore, we can follow the same procedure to construct a refute set for Problem [1] and then use Positivstellensatz to turn the problem into a feasibility problem similar to (6), which can then be formed into an ordinary nonlinear program.

**V. METHOD**

In this section, we will introduce four steps to solve Problem [1] leveraging Theorem [1].

**A. The Local Manifold Positiveness Problem**

Since Theorem [1] applies to a set of equalities, inequations, and inequalities, we need to unroll $\min_{u \in U} \hat{\phi}(x, u)$ and get rid of min operator. Firstly, the equivalent form of $\min_{u \in U} \hat{\phi}(x, u)$ is summarized as following:

$$\min_{u \in U} \hat{\phi}(x, u) = \min_{u \in U} \frac{\partial \hat{\phi}}{\partial x} f(x) + \frac{\partial \hat{\phi}}{\partial x} g(x) u \tag{7}$$

$$= \min_{u \in U} L_f \hat{\phi} + \sum_{i=1}^{n_u} L_g \hat{\phi}^i u^i,$$

where $L_g \hat{\phi}^i$ and $u^i$ denote the $i$-th dimension of $L_g \hat{\phi}$ and $u$, respectively. (7) can be rewritten as

$$\min_{u \in U} \hat{\phi}(x, u) = L_f \hat{\phi} + \sum_{i=1}^{n_u} L_g \hat{\phi}^i \|L_g \hat{\phi}^i\| \left(u_{i_{\min}}^i, u_{i_{\max}}^i\right), \tag{8}$$

where we define operator $\| \cdot \|$ as $\|A\{B, C\} = B$ if $A \geq 0$, and $\|A\{B, C\} = C$ if $A < 0$. Therefore, Problem [1] becomes

**Problem 2 (Local Manifold Positiveness):** Find

$$\min_{u \in U} \hat{\phi}(x, u) = L_f \hat{\phi} + \sum_{i=1}^{n_u} L_g \hat{\phi}^i \|L_g \hat{\phi}^i\| \left(u_{i_{\min}}^i, u_{i_{\max}}^i\right) < 0, \tag{9}$$
where local denotes the state space $X$ which is a subset of $\mathbb{R}^n$, and manifold denotes we are considering the states on the manifold defined by $\phi_\Theta = 0$.

Remark 2: The state-of-the-art SOSP-based safety index synthesis methods try to synthesis $\phi_\Theta$, such that $\forall x \in \mathbb{R}^n, \exists u \in \mathcal{U} \phi_\Theta (x,u) < 0$, which is a global positiveness constraint. Our formulation only requires the existence of safe control on a local set of critical states, i.e. local manifold positiveness constraint. Therefore, our method considers a larger solution space; hence it is easier to find a solution, and the synthesized safety index is less conservative.

B. The Refute Problem

We can show that the local manifold positiveness constraint in Problem 2 is satisfied by showing its refute set is empty. The refute set is constructed as:

$$\begin{align*}
L_f \phi + \sum_{i=1}^{u} L_g \phi[i] u_{\min}[i], u_{\max}[i] & \geq 0 \\
x \in X \\
\phi_\Theta = 0
\end{align*}$$

(10)

Denote $N, M \subseteq \{1, 2, \ldots, n_u\}$, $N + M = \{1, 2, \ldots, n_u\}$ and $N \cap M = \emptyset$. We also denote $\forall i \in N, L_g \phi[i] \geq 0$, and $\forall i \in M, L_g \phi[i] \leq 0$. Then, (10) corresponds to $2^{n_u}$ instances of refute set, since each $L_g \phi[i]$ can be either negative or nonnegative, yielding $2^{n_u}$ combinations of $|N|, |M|$. Then each instance of (10) can be rewritten as:

$$\begin{align*}
\gamma^*_i := L_f \phi + \sum_{i \in N} L_g \phi[i], u_{\min}[i], u_{\max}[i] \\
\gamma^*_i := S(x)_i \geq 0, i = 1, 2, \ldots, n_S \\
\gamma^*_i = 0, i \notin N \\
\gamma^*_i := -L_g \phi[i] \geq 0, i \in M \\
\zeta^* := \phi_\Theta = 0
\end{align*}$$

(11)

Theorem 1 enables us to turn the emptiness problem for (11) to a feasibility problem similar to (6). As a result, we can show (11) is empty using the following condition:

$$\begin{align*}
\exists q_i \in \mathbb{R}[x], \exists p_i \in \text{SOS}, \forall i, \\
\text{s.t. } \gamma = p_0 + p_1 \gamma^*_0 + \ldots + p_r \gamma^*_N \\
+ p_{01} \gamma^*_0 \gamma^*_1 + \ldots + p_{012} \ldots \gamma^*_N \gamma^*_N, \\
\zeta = q_i \zeta^*, \\
\gamma + \zeta + 1 = 0,
\end{align*}$$

(12)

where $N = n_S + |N| + |M|$. Therefore, the Problem 2 can be turned into the following equivalent problem:

Problem 3 (Refute): find $\Theta = \{k_1, k_2, \ldots, k_n\}$, such that

$$\forall N, M, (12) \text{ holds}$$

(13)

Remark 3: It is noteworthy that Theorem 1 requires $\gamma^*$ and $\eta^*$ to be polynomials, which limits the applicability of current SOSP-based safety index synthesis methods. On the other hand, as long as the system dynamics satisfy Definition 1, our proposed method can be applied.

Remark 4: Note that since $L_g \phi$ depends on $x$ and state space $X$ is bounded, not all combinations of $\{N, M\}$ are possible. For those impossible combinations of $\{N, M\}$, (11) is directly empty, which further eliminates the number of (11) to be considered.

C. The Nonlinear Programming Problem

To efficiently search for the existence of SOS polynomials $\{p_i\}$ and polynomials $q_i$, we set $p_i$ to be a positive scalar $\alpha_i$ for all $i \geq 1$ and $q_1$ to be a scalar $\beta_1$. Hence, a simplified condition of (12) can be defined:

$$\begin{align*}
\exists \beta_1 \in \mathbb{R}, \exists \alpha_i \geq 0, \forall i \geq 1, \\
\text{s.t. } p_0 = -\alpha_1 \gamma^*_0 - \ldots - \alpha_s \gamma^*_N - \alpha_0 \gamma^*_1 - \ldots - \alpha_{012} \ldots \gamma^*_N - \beta_1 \zeta^* - 1 \in \text{SOS},
\end{align*}$$

(14)

By searching for limited types of SOS polynomials, the existence of $\beta_1, \alpha_i$ satisfying (14) is sufficient to satisfy the constraints in (12).

Denote $\Omega = [\beta_1, \alpha_1, \alpha_2, \ldots, \alpha_{012} \ldots]$ as the decision vector for (14). To solve (14), suppose the degree of $p_0$ is $2d$, and we first do a sum-of-squares decomposition of $p_0$ such that $p_0 = Y^T Q^*(\Omega, \Omega)$, where $Q^*$ is symmetric and $Y = [x^{[1]} \ x^{[2]} \ \ldots \ x^{[n]} \ x^{[1]} \ x^{[2]} \ \ldots \ (x^{[n]}])^d$ and $x^{[i]}$ is the entry in $x$. Specifically, for off-diagonal terms $Q_{ij}$ as the element of $Q^*$ at $i$-th row and $j$-th column ($i \neq j$), assuming that the coefficient of the term $Y_i Y_j$ in $p_0$ is $w_{ij}$, we set $Q_{ij}^{\ast} = 1$. With decomposed $Q^*$, the condition of (14) can be rewritten as:

$$\det(Q^*(\Omega, \Omega)_k) > 0, \forall k,$$

(15)

where $Q^*(\Omega, \Omega)_k$ denotes the $k \times k$ submatrix consisting of the first $k$ rows and columns of $Q^*(\Omega, \Omega)$. According to Problem 3 there are $2^{n_u}$ sets of condition (12) and hence $2^{n_u}$ sets of condition (14). Denote the $\Omega_j = [\beta_1^j, \alpha_1^j, \alpha_2^j, \ldots, \alpha_{012}^j \ldots]$ as the decision vector for $j$-th set of condition (14). With (15), Problem 3 can be rewritten as:

Problem 4 (Nonlinear Programming): Find $[\Theta, \Omega_1, \Omega_2, \ldots, \Omega_{2^{n_u}}]$, such that

$$\begin{align*}
\det(Q^*(\Omega, \Omega)_k) > 0, \forall k, \forall j, \\
\alpha^j_i > 0, \beta^j \in \mathbb{R}, \forall i, \forall j.
\end{align*}$$

(16)

Note that Problem 4 is a nonlinear programming problem without any objective. Hence, any arbitrary objective can be added to Problem 4 and it can be solved by off-the-shelf nonlinear programming solvers.

Remark 5: Note that the state-of-the-art SOSP-based safety index synthesis methods need to solve multiple sets of SOSP iteratively. On the other hand, Problem 4 demonstrates that our method just needs to solve one nonlinear program problem. However, since Problem 4 involves enumerating at most $2^{n_u}$ combinations of constraint sets, which might not scale to high-dimensional systems (the scalability depends on the bound of $X$).

VI. Properties of General Safety Index Design

This section proves that we can obtain a feasible safety index design by solving Problem 4. The main result is summarized in the following theorem.
Theorem 2 (Feasibility of General Safety Index Design): The solution of Problem 4 is also a solution of Problem 1.

Proof: Denote the solution of Problem 4 as \( \Theta_3 \), the solution of Problem 3 as \( \Theta_2 \), the solution of Problem 2 as \( \Theta_1 \), and the solution of Problem 1 as \( \Theta^o \).

Relationship between \( \Theta_1 \) and \( \Theta^o \): According to (7), the following condition holds

\[
\min_{u \in \mathcal{U}} \phi(x, u) = L_f \hat{\phi} + \sum_{i=1}^{n_u} \min_{u[i]} L_g \phi[i] u[i],
\]

where \( \min_{u \in \mathcal{U}} L_g \phi[i] u[i] = L_g \phi[i] \sup_{u \in \mathcal{U}} u[i] \) if \( L_g \phi[i] < 0 \), and \( \min_{u \in \mathcal{U}} L_g \phi[i] u[i] = L_g \phi[i] \inf_{u \in \mathcal{U}} u[i] \) if \( L_g \phi[i] \geq 0 \). Hence, the condition \( \min_{u \in \mathcal{U}} \phi(x, u) \) is equivalent to the condition \( L_f \hat{\phi} + \sum_{i=1}^{n_u} L_g \phi[i] \left[ u[i] \right]_{\min, \max} \right) \). So \( \Theta_1 \) is an instance of \( \Theta^o \).

Relationship between \( \Theta_1 \) and \( \Theta_2 \): For the constraint \( \forall x \in \mathcal{X}, \phi = 0 \), \( L_f \hat{\phi} + \sum_{i=1}^{n_u} \min_{u[i]} L_g \phi[i] u[i] < 0 \), its refute certification is \( \forall x \in \mathcal{X}, \phi = 0 \), \( L_f \hat{\phi} + \sum_{i=1}^{n_u} \min_{u[i]} L_g \phi[i] u[i] < 0 \). By introducing auxiliary variables, the refute certifications of the constraints in Problem 2 can be written as (11). By Theorem 1 definition 2 for \( \Gamma \) and definition for \( \Phi \) [14], we know (11) is equivalent to (12). Hence, Problem 2 is equivalent to Problem 3, which indicates \( \Theta_2 \) is an instance of \( \Theta_1 \).

Relationship between \( \Theta_2 \) and \( \Theta_3 \): The equivalent condition to (14) is that \( Q^* \) is positive semidefinite [14]. The condition (15) proves that \( Q^* \) is positive definite according to Sylvester’s criterion. Hence, the decision vectors \( \{\Omega_1, \Omega_2, \ldots, \Omega_{n_u}\} \) satisfying condition (15) also satisfy condition (14).

The simplest SOS is a positive constant scalar, i.e., \( \alpha \geq 0 \) in SOS. Similarly, the simplest polynomial is a constant scalar, i.e., \( \beta \in \mathbb{R} \). Hence, by substituting \( p_i \) with \( \alpha_i \geq 0 \) for \( i = 1, 2, \ldots \), and substituting \( q_i \) with \( \beta_i \in \mathbb{R} \), condition (14) can be rewritten as (12). Therefore, the existence of \( \{\Omega_1, \Omega_2, \ldots, \Omega_{n_u}\} \) satisfying the conditions in Problem 4 implies the satisfaction of conditions in Problem 3 Therefore, \( \Theta_3 \) is an instance of \( \Theta_2 \).

In summary, \( \Theta_3 \) is an instance of \( \Theta_2 \), hence an instance of \( \Theta_1 \), and hence an instance of \( \Theta^o \), which verifies the claim.

VII. NUMERICAL STUDY

A. Robot Arm Numerical Study Setup

We evaluate our method in 2D robot arm systems with different degrees of freedom (DOF). The experimental platform is illustrated in Fig. 2. The link length of the robot is 1 meter. The obstacle is set as a half plane 0.5 m x n meter away from the robot base with nDOF.

The state space includes 1) each joint angle and 2) each joint velocity. Each joint angle is bounded by a subinterval \([\frac{\pi}{2}, \frac{3\pi}{2}]\) and each joint velocity is bounded within \([-1, 1]\). The control inputs are accelerations of each joint. The acceleration is limited within \([-1, 1]\).

B. Robot Arm Running Example

In this subsection, we introduce a running example showing 1DOF robot arm safety index design via SOS programming. Consider 1) robot state \( x = [\theta, \dot{\theta}] \), where \( \theta \in [\pi/3, 2\pi/3], \dot{\theta} \in [-1, 1] \); 2) robot control \( u = [\theta] \), where \( \dot{\theta} \in [-1, 1] \). Consider the user-defined safety index as \( \phi_0 = \cos \theta - 0.5 \). Then, safety index becomes \( \phi = \cos \theta - 0.5 - k \sin \dot{\theta} \), with \( k = -\sin \theta - k \cos \theta \dot{\theta}^2 - k \sin \dot{\theta} \). Since \( \sin \theta > \frac{\sqrt{3}}{2} \), the minimum \( \phi \) is achieved when \( \dot{\theta} = 1 \), i.e., \( \min_{\theta} \phi = -\sin \theta - k \cos \theta \dot{\theta}^2 - k \sin \dot{\theta} \). So the fundamental condition for nonempty set of safe control is:

\[
\begin{align*}
    f_1 &= -\sin \theta - k \cos \theta \dot{\theta}^2 - k \sin \dot{\theta} < 0 \\
    f_2 &= \theta - \frac{\pi}{2} \geq 0 \\
    f_3 &= 2\pi - \theta \geq 0 \\
    f_4 &= 1 - \dot{\theta}^2 \geq 0 \\
    h_1 &= \cos \theta - 0.5 - k \sin \dot{\theta} = 0
\end{align*}
\]

Next, the equivalent refute set for (18) can be constructed by replacing \( f_1 < 0 \) in (18) with \( f_1 \geq 0 \), which indicates that \( \dot{\theta} \) hits zero, no \( x \in \mathcal{X} \) would satisfy \( \min_{\theta} \phi > 0 \). By projecting the non-polynomial terms of (15) into higher dimensional polynomial manifold, i.e. substituting \( \sin \theta = x_1, \cos \theta = x_2, x_3 = x_3 \), the refute set becomes:

\[
\begin{align*}
    f_1 &= -x_1 x_3 - k x_2 x_3^2 - k x_1 \geq 0 \\
    f_2 &= x_1 \geq \frac{\sqrt{3}}{2} \\
    f_3 &= 1 - x_1 \geq 0 \\
    f_4 &= 1 - x_3^2 \geq 0 \\
    h_1 &= x_2 - 0.5 - k x_1 x_3 = 0 \\
    h_2 &= x_1^2 + x_3^2 - 1 \leq 0
\end{align*}
\]

According to Theorem 1, the emptiness of (19) is equivalent to the existence of \( p_0, p_1, p_2, p_3, p_4 \in \text{SOS polynomials}, q_1, q_2 \in \text{polynomials} \), such that:

\[
p_0 + p_1 f_1 + p_2 f_2 + p_3 f_3 + p_4 f_4 + q_1 h_1 + q_2 h_2 + 1 = 0
\]

By considering a reduced parameter space, the emptiness of (19) is equivalent to the existence of \( p_1, p_2, p_3, p_4, q_1, q_2 \) in \( \mathbb{R}^+ \),

\[
p_0 = -(p_1 f_1 + p_2 f_2 + p_3 f_3 + p_4 f_4 + q_1 h_1 + q_2 h_2 + 1)
\]

is a sum of squares polynomial, where \([p_1, p_2, p_3, p_4, q_1, q_2]\) can be searched via nonlinear programming.

To design safety index design via SOS for higher DOF robot arm, we need to construct \( n \) sets of inequalities and equalities similar to (19) for the refute set of nDOF case, due to the fact that all joints are independent with each other.

C. Robot Arm Results

To solve for safety index design, we first provide reference \([\Theta, \Omega_1, \Omega_2, \ldots, \Omega_{2n_u}]\) via random sampling, and then solve Problem 4 using MATLAB \( \text{fmincon} \) function via interior point solver. To evaluate safety index design, we randomly initialize the robot, and use safe set algorithm [9] to safeguard the robot for 2000 time steps, where the robot


### Table I: The performance of safety index design via SOS programming with 1000 random seeds (Robot Arms). Average computing time, validness, variance, and the changes of solving a feasible solution are reported.

| Number of robot arm links | Time (s) | Validity (%) | Variance | Feasibility (%) |
|---------------------------|----------|--------------|----------|-----------------|
| 2DOF                      | 0.160    | 100          | 0.118    | 99.5            |
| 4DOF                      | 0.180    | 100          | 0.042    | 95.6            |
| 6DOF                      | 0.196    | 100          | 0.147    | 92.2            |
| 10DOF                     | 0.303    | 100          | 0.247    | 91.6            |
| 14DOF                     | 0.422    | 100          | 0.342    | 34.7            |

**VIII. CONCLUSIONS**

This paper proposed a framework for synthesizing the safety index for general control systems under control limits using sum-of-squares programming. Our approach leverages Positivstellensatz theorem to ensure the non-emptiness of safe control. The experimental results show that the synthesized safety index guarantees safety and our method is effective even in high-dimensional robot systems. The proposed method is limited to white-box analytical dynamics. Future research could focus on safety index synthesize for black-box dynamics.

### References

[1] T. Wei and C. Liu, “Safe control algorithms using energy functions: A unified framework, benchmark, and new directions,” in 2019 IEEE 58th Conference on Decision and Control (CDC), Nice, France. IEEE, 2019, pp. 238–243.

[2] Z. Zhao, T. He, and C. Liu, “Model-free safe control for zero-violation reinforcement learning,” in 5th Annual Conference on Robot Learning, London, UK, 2021.

[3] W. S. Cortez and D. V. Dimarogonas, “Safe-by-design control for Euler-Lagrange systems,” arXiv preprint arXiv:2009.03767, 2020.

[4] A. Clark, “Verification and synthesis of control barrier functions,” in 60th Conference on Decision and Control (CDC), Austin, Texas, USA. IEEE, 2021.

[5] T. Wei and C. Liu, “Safe control with neural network dynamic models,” in Learning for Dynamics and Control Conference, Palo Alto, CA, USA, 2022.

[6] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in 18th European control conference, Naples, Italy. IEEE, 2019.

[7] W. Zhao, S. He, and C. Liu, “Provably safe tolerance estimation for robot arms via sum-of-squares programming,” IEEE Control Systems Letters, 2022.

[8] S. Liu, J. Dolan, and C. Liu, “Safe control under input saturation with neural control barrier functions,” in 6th Annual Conference on Robot Learning, Auckland, NZ, 2022.

[9] C. Liu and M. Tomizuka, “Control in a safe set: Addressing safety in human-robot interactions,” in ASME 2014 Dynamic Systems and Control Conference, San Antonio, Texas, USA. ASME Digital Collection, 2014.

[10] H. Ma, C. Liu, S. E. Li, S. Zheng, and J. Chen, “Joint synthesis of safety certificate and safe control policy using constrained reinforcement learning,” in Learning for Dynamics and Control Conference, PMLR, 2022, pp. 97–109.

[11] A. Clark, “Control barrier functions for stochastic systems,” Automatica, vol. 130, p. 109688, 2021.

[12] C. Dawson, Z. Qin, S. Gao, and C. Fan, “Safe nonlinear control using robust neural Lyapunov-barrier functions,” in Conference on Robot Learning, London, UK. PMLR, 2022, pp. 1724–1735.

[13] J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, “Robust control barrier–value functions for safety-critical control,” in 2021 60th IEEE Conference on Decision and Control (CDC), Austin, Texas, USA. IEEE, 2021, pp. 6814–6821.

[14] P. A. Parrilo, “Semidefinite programming relaxations for semialgebraic problems,” Mathematical programming, vol. 96, pp. 293–320, 2003.