2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Search-based and Stochastic Solutions to the Zonotope and Ellipsotope Containment Problems

Adrian Kulmburg, Ivan Brkan, and Matthias Althoff

***Abstract*— We introduce new techniques to check whether a zonotope is contained in another zonotope. This fundamental problem in control theory has many applications, such as the verification of invariant sets, formal verification of controllers, and fault detection. Our first method uses a search-based vertex enumeration to quickly and efficiently check containment. We also propose two stochastic methods that are able to rapidly disprove or confirm containment with a certain probability. Furthermore, we generalize the first approach to the case where the circumbody is an ellipsotope and generalize the stochastic methods to the case where both the inbody and the circumbody are ellipsotopes. We conclude by comparing the efficiency of our algorithms to currently available ones.**

I. Introduction

The zonotope containment problem is a classical problem in optimization that asks whether a zonotope is contained within another zonotope. A zonotope is a type of convex polytope that can be described as the Minkowski sum of line segments. The containment problem is required for many applications, such as robust control [1], controller synthesis [2], and conformance checking [3]. Unfortunately, the zonotope containment problem is NP-hard [4], which helps explain why only relatively simple methods (summarized in [4]) are known in the literature, though approximative approaches have also been developed in [5], and to some extent in [3].

The high runtime is the main drawback of existing methods. Workarounds have been explored in [5] by presenting a necessary but not sufficient condition for containment. While, in practice, this algorithm often predicts correctly whether a zonotope is contained within another, in cases where this algorithm fails to confirm containment (in which case containment may still hold), there is no way to refine the analysis (for instance, at the cost of additional runtime) to disprove containment. Therefore, there is a need for new strategies that can either quickly disprove containment or have a tunable accuracy with respect to some metric we are about to define.

This article introduces three novel techniques to check for containment: The first is based on a search through a binary tree and will be presented in Section III, while the other two are stochastic algorithms that can either disprove containment or confirm it with a certain probability, as we will discuss in Section IV. This realizes a more flexible approximation since the runtime is directly proportional to the number of samples the user wants to employ. We evaluate their performance in Section V.

Moreover, we generalize our methods for zonotopes so that they can also be applied, in most instances, to the containment of ellipsotopes [6], which generalize zonotopes.

This work has been financially supported by the European Commission project justITSELF under grant number 817629.

The authors are with the Chair of Robotics, Artificial Intelligence and Real-time Systems, Technische Universität München, Boltzmannstr. 3, 85748 Garching b. München, Germany.
Email: {adrian.kulmburg,ivan.brkan,althoff}@tum.de

II. Preliminaries

A. Basic Notation

A letter with an arrow (e.g., $\vec{v}$) represents a **vector** in $\mathbb{R}^n$, while **matrices** in $\mathbb{R}^{n\times m}$ are denoted by bold, underlined letters (e.g., $\underline{\boldsymbol{M}}$). The vectors $\vec{e}_i \in \mathbb{R}^n$ for $i = 1, ..., n$ are the **canonical basis vectors** of $\mathbb{R}^n$. For a vector $\vec{v} \in \mathbb{R}^n$, v_i for $i = 1, ..., n$ refers to the coordinates of $\vec{v}$ with respect to the canonical basis. If $\underline{\boldsymbol{A}}, \underline{\boldsymbol{B}}$ are matrices (or vectors), then $[\underline{\boldsymbol{A}}, \underline{\boldsymbol{B}}]$ is the **horizontal**, $\begin{bmatrix}\underline{\boldsymbol{A}}\\ \underline{\boldsymbol{B}}\end{bmatrix}$ the **vertical concatenation** of $\underline{\boldsymbol{A}}$ and $\underline{\boldsymbol{B}}$ (assuming the dimensions of $\underline{\boldsymbol{A}}$ and $\underline{\boldsymbol{B}}$ match). For a vector $\vec{v}$ and $p \in [1, \infty)$, $\|\vec{v}\|_p := \sqrt[p]{|v_1|^p + ... + |v_n|^p}$ is the **p-norm** of $\vec{v}$, for $p = \infty$, $\|\vec{v}\|_\infty := \max_i |v_i|$. We denote by $\mathcal{B}_p$ the **unit ball** with respect to the p-norm. For vectors $\vec{v}, \vec{w} \in \mathbb{R}^n$, their (Euclidean) **inner product** is written as $\vec{v}^\top \vec{w} = \langle \vec{v}, \vec{w} \rangle$, and their **Hadamard product** as $\vec{v} \circ \vec{w}$, i.e., $(\vec{v} \circ \vec{w})_i = v_i w_i$. For a probability space Ω and some measurable subset $S \subseteq E$ of some measurable space E with a random variable $\mathsf{s} : \Omega \to E$, $\mathbb{P}(\mathsf{s} \in S)$ is the **probability** that $\mathsf{s} \in S$.

B. Duality

If $\|\cdot\|$ is a norm on $\mathbb{R}^n$, we denote by $\|\cdot\|^*$ its dual norm (see [7, Chapter 1.3]), which is defined as

$$\|\vec{y}\|^* = \sup_{\|\vec{x}\| \leq 1} \vec{y}^\top \vec{x}. \tag{1}$$

For the vector p-norms, it is a common exercise to prove

$$\|\vec{x}\|_p^* = \|\vec{x}\|_{p^*}, \tag{2}$$

where p^* is the Hölder-conjugate of p, i.e., $\frac{1}{p} + \frac{1}{p^*} = 1$. For sets $S \subseteq \mathbb{R}^n$, the dual set is defined as

$$S^* := \left\{ \vec{y} \in \mathbb{R}^n \;\middle|\; \sup_{\vec{x} \in S} \vec{y}^\top \vec{x} \leq 1 \right\}. \tag{3}$$

C. Ellipsotopes, Zonotopes, and Ellipsoids

Let us now introduce our primary object of study, namely ellipsotopes:

Definition 2.1 (Ellipsotopes, see [6, Definition 2]): For $m, n \in \mathbb{N}$, let $\underline{\boldsymbol{G}} \in \mathbb{R}^{n\times m}$ be a matrix, $\vec{c} \in \mathbb{R}^n$ a vector, and $p \in [1, \infty]$. Then

$$E_p(\underline{\boldsymbol{G}}, \vec{c}) = \left\{ \vec{c} + \underline{\boldsymbol{G}}\vec{\alpha} \;\middle|\; \|\vec{\alpha}\|_p \leq 1 \right\} \tag{4}$$

is called a (basic) *ellipsotope* (or p-*ellipsotope*). The vector $\vec{c}$ is the *center*, the matrix $\underline{\boldsymbol{G}}$ the *generator matrix*, and column vectors $\vec{g}_1, ..., \vec{g}_m$ of $\underline{\boldsymbol{G}}$ are *generators*. We will use the shorthand $E_p(\underline{\boldsymbol{G}})$ to denote the p-ellipsotope centered at the origin with generator matrix $\underline{\boldsymbol{G}}$. Ellipsotopes with $p = \infty$ are called *zonotopes*. We will use the shorthand $Z(\underline{\boldsymbol{G}})$ to denote the zonotope centered at the origin, with generator matrix $\underline{\boldsymbol{G}}$.

Since ellipsotopes are convex and centrally symmetric, they induce a norm:

Proposition 2.2 (Ellipsotope norm): Let $m \geq n$, and suppose $\underline{\boldsymbol{G}} \in \mathbb{R}^{n \times m}$ has full rank. Then for any $p \in [1, \infty]$ the function

$$\begin{aligned} \mathbb{R}^n &\to [0, \infty) \\ \vec{x} &\mapsto \min_{\underline{\boldsymbol{G}}\vec{\alpha} = \vec{x}} \|\vec{\alpha}\|_p \end{aligned} \tag{5}$$

defines a norm on $\mathbb{R}^n$, which we denote by $\|\vec{x}\|_{E_p(\underline{\boldsymbol{G}})}$. Its unit circle coincides with $E_p(\underline{\boldsymbol{G}})$. If $p = \infty$, we instead denote this norm by $\|\vec{x}\|_{Z(\underline{\boldsymbol{G}})}$.

A proof of Proposition 2.2 for zonotopes is shown in [4, Section 3.2.]. We leave the generalization to ellipsotopes as an easy exercise for the reader. Note that evaluating $\|\vec{x}\|_{E_p(\underline{\boldsymbol{G}})}$ involves solving a convex minimization problem, which can be done in polynomial time (with respect to the size of $\underline{\boldsymbol{G}}$) up to arbitrary accuracy as described in [8, Chapter 11].

D. The Containment Problem

We now turn towards the problem of checking whether an ellipsotope $\breve{E}$ (called the **inbody**) is contained within another ellipsotope $\widehat{E}$ (called the **circumbody**). Suppose $\breve{E} = E_p(\underline{\boldsymbol{G}}, \vec{c})$ and $\widehat{E} = E_q(\underline{\boldsymbol{H}}, \vec{d})$ for $p, q \in [1, \infty]$, $\underline{\boldsymbol{G}} \in \mathbb{R}^{n \times m}$, $\vec{c} \in \mathbb{R}^n$, $\underline{\boldsymbol{H}} \in \mathbb{R}^{n \times l}$, and $\vec{d} \in \mathbb{R}^n$. The generalization of the containment problem of zonotopes [4, Equation (18)] for ellipsotopes is as follows: $\breve{E} \subseteq \widehat{E}$ holds if and only if $r(\breve{E}, \widehat{E}) \leq 1$, where

$$\begin{aligned} r(\breve{E}, \widehat{E}) &:= \max_{\|\vec{x}\|_{E_p(\underline{\boldsymbol{G}})} \leq 1} \left\| \vec{x} + \vec{c} - \vec{d} \right\|_{E_q(\underline{\boldsymbol{H}})} \\ &= \max_{\|\vec{\alpha}\|_p \leq 1} \min_{\underline{\boldsymbol{H}}\vec{\beta} = \underline{\boldsymbol{G}}\vec{\alpha} + \vec{c} - \vec{d}} \|\vec{\beta}\|_q \end{aligned} \tag{6}$$

As we have shown in [9, Corollary 3.5.], the dual formulation of (6) is

$$r(\breve{E}, \widehat{E}) = \max_{\|\underline{\boldsymbol{H}}^\top \vec{x}\|_{q^*} \leq 1} \left\| \underline{\boldsymbol{G}}^\top \vec{x} \right\|_{p^*} + \vec{x}^\top (\vec{c} - \vec{d}). \tag{7}$$

III. Search-Based Techniques

In this section, we only consider the case where $\breve{E}$ is a zonotope $\breve{Z}$, since the next few arguments fail for general ellipsotopes, though $\widehat{E}$ can still be an arbitrary ellipsotope. In this case, according to the Bauer maximum principle, the maximum of (6) is attained at one of the vertices of $\{\vec{\alpha} \in \mathbb{R}^m \mid \|\vec{\alpha}\|_\infty \leq 1\}$, i.e., at a point $\vec{\alpha} \in \{-1, +1\}^m$. We can thus iteratively choose either $\alpha_i = -1$ or $\alpha_i = +1$, which can be seen as a depth-first search through a binary search tree, where to each node at depth k for $0 \leq k \leq m$ we associate a $\vec{\sigma} \in \{\pm 1\}^k$, which corresponds to a choice of the first k coefficients α_i. We define the value of such a node as

$$\widetilde{r}_k(\vec{\sigma}) := \left\| \vec{c} - \vec{d} + \sum_{i=1}^{k} \vec{g}_i \sigma_i \right\|_{E_q(\underline{\boldsymbol{H}})}. \tag{8}$$

In each iteration, we greedily choose α_{k+1} to maximize the total norm, which gives rise to Algorithm 1, where the operation $\texttt{push}(\text{List}, x)$ adds an element x to a list, $\texttt{pop}(\text{List})$ returns the last element of the list before removing it from the list, and $\texttt{dim}(\vec{x})$ for $\vec{x} \in \mathbb{R}^k$ returns the dimension of $\vec{x}$ (i.e., k). We use the convention that a 0-dimensional vector is just the empty set $\emptyset$. Note that by definition of the empty set, $\emptyset = ()$ (where $()$ is the empty tuple) but $\emptyset \neq (\emptyset)$.

Algorithm 1 Depth-First Vertex Enumeration

Input: Zonotope $\breve{Z}$ with generators $\vec{g}_1, ..., \vec{g}_m$ and center $\vec{c}$, ellipsotope $\widehat{E} = E_q(\underline{\boldsymbol{H}}, \vec{d})$.
Output: True if $\breve{Z} \subseteq \widehat{E}$, False otherwise.

Sort $\vec{g}_1, ..., \vec{g}_m$ according to $\|\vec{g}_i\|_{E_q(\underline{\boldsymbol{H}})}$, in descending order
List $\leftarrow (\emptyset)$
$M \leftarrow \|\vec{c} - \vec{d}\|_{E_q(\underline{\boldsymbol{H}})}$
if $M > 1$ **then**
  return False
end if
while List $\neq \emptyset$ **do**
  $\vec{\sigma} \leftarrow \texttt{pop}(\text{List})$
  $k \leftarrow \texttt{dim}(\vec{\sigma})$
  if $h_k(\vec{\sigma}) \leq 1$ **then**
    continue
  end if
  Set $\vec{\sigma}_+ \leftarrow \begin{bmatrix} \vec{\sigma} \\ +1 \end{bmatrix}$ and $\vec{\sigma}_- \leftarrow \begin{bmatrix} \vec{\sigma} \\ -1 \end{bmatrix}$
  if $\widetilde{r}_{k+1}(\vec{\sigma}_+) \geq \widetilde{r}_{k+1}(\vec{\sigma}_-)$ **then**
    $\texttt{push}(\text{List}, \vec{\sigma}_-)$
    $\texttt{push}(\text{List}, \vec{\sigma}_+)$
  else
    $\texttt{push}(\text{List}, \vec{\sigma}_+)$
    $\texttt{push}(\text{List}, \vec{\sigma}_-)$
  end if
  $M \leftarrow \max\{\widetilde{r}_{k+1}(\vec{\sigma}_-), \widetilde{r}_{k+1}(\vec{\sigma}_+)\}$
  if $M > 1$ **then**
    return False
  end if
end while
return True

Algorithm 1 starts by sorting the generators according to their $\|\cdot\|_{E_q(\underline{\boldsymbol{H}})}$-norm, as generators with a higher $\|\cdot\|_{E_q(\underline{\boldsymbol{H}})}$-norm have a higher chance to contribute more to the total value of $r(\breve{Z}, \widehat{E})$. Additionally, we can eliminate certain nodes that have no chance of leading to the maximum: suppose that in an arbitrary iteration, we want to continue with the node given by $\vec{\sigma} \in \{-1, +1\}^k$ for $0 \leq k < m$. For each sub-node $\vec{\sigma}' \in \{-1, +1\}^{k'}$ for $k < k' \leq m$, $\sigma'_i = \sigma_i$ holds for each of the first $1 \leq i \leq k$. By the

triangle inequality we have

$$\begin{aligned}\widetilde{r}_{k'}(\vec{\sigma}') &= \left\| \vec{c} - \vec{d} + \sum_{i=1}^{k'} \vec{g}_i \sigma_i' \right\|_{E_q(\underline{\boldsymbol{H}})} \\ &\leq \left\| \vec{c} - \vec{d} + \sum_{i=1}^{k} \vec{g}_i \sigma_i \right\|_{E_q(\underline{\boldsymbol{H}})} + \sum_{i=k+1}^{k'} \|\vec{g}_i\|_{E_q(\underline{\boldsymbol{H}})} \\ &= \widetilde{r}_k(\vec{\sigma}) + \sum_{i=k+1}^{k'} \|\vec{g}_i\|_{E_q(\underline{\boldsymbol{H}})} .\end{aligned}$$

Consequently, we may consider the value

$$h_k(\vec{\sigma}) := \widetilde{r}_k(\vec{\sigma}) + \sum_{i=k+1}^{m} \|\vec{g}_i\|_{E_q(\underline{\boldsymbol{H}})} \tag{9}$$

as a heuristic to eliminate certain nodes, since if $h_k(\vec{\sigma}) \leq 1$, there is no chance for any sub-node to achieve a greater value than 1. Since the values of $\|\vec{g}_i\|_{E_q(\underline{\boldsymbol{H}})}$ are already computed in Line 1, computing $h_k(\vec{\sigma})$ boils down to summing already known terms and thus does not significantly contribute to the runtime.

Finally, since we are primarily concerned with determining whether $\breve{Z} \subseteq \widehat{E}$ (and not necessarily computing the exact value of (6)), we can interrupt the algorithm whenever a value of M has been found such that $M > 1$, in which case containment has been disproven. Combined with the use of the heuristic $h_k(\vec{\sigma})$, this enables Algorithm 1 to be particularly fast whenever $\breve{Z}$ is significantly smaller or larger than $\widehat{E}$, and only in cases where most vertices of $\breve{Z}$ are close to the boundary of $\widehat{E}$, it may happen that Algorithm 1 needs to check a large number of vertices. As we will see in Section V, this leads to a significantly better runtime for most instances of the containment problem, even though the worst-case runtime is still exponential in m.

IV. Sampling-Based Techniques

We now introduce methods that solve the containment problem via random sampling. These algorithms are typically fast and have a runtime that is easy to control (it suffices to set a maximum number of samples). However, they do not always yield guaranteed results. For the remainder of this section, $N \in \mathbb{N}$ denotes the number of random samples used for the algorithms.

A. Naive Approaches for the Zonotope Containment Problem

Before introducing more refined stochastic methods, we present some deliberately simple sampling-based approaches to intuitively introduce our two main sampling techniques.

1) Naive Vertex Sampling: This method only works if $\breve{E}$ is a zonotope $\breve{Z}$ (though $\widehat{E}$ can be an arbitrary ellipsotope), and is based, again, on the Bauer maximum principle: Since the maximum in (6) is reached at one of the vertices of the convex domain $\{\vec{\alpha} \in \mathbb{R}^m \mid \|\vec{\alpha}\|_\infty \leq 1\}$, it would suffice to randomly sample points $\vec{\alpha} \in \{-1, +1\}^m$ and take the maximum of the function $\|\underline{\boldsymbol{G}}\vec{\alpha} + \vec{c} - \vec{d}\|_{E_q(\underline{\boldsymbol{H}})}$ over all sampled points. Specifically, if N points $\vec{\mathfrak{a}}_1, ..., \vec{\mathfrak{a}}_N$ are sampled uniformly in $\{-1, +1\}^m$, this yields an approximation

$$\mathfrak{r}_{\text{vs}} := \max_{1 \leq i \leq N} \left\| \underline{\boldsymbol{G}}\vec{\mathfrak{a}}_i + \vec{c} - \vec{d} \right\|_{E_q(\underline{\boldsymbol{H}})}, \tag{10}$$

such that $\mathfrak{r}_{\text{vs}} \leq r(\breve{Z}, \widehat{E})$. We will call this approach *naive vertex sampling*.

2) Naive Halfspace Sampling: This approach only works if $\widehat{E}$ is a zonotope $\widehat{Z}$ (though $\breve{E}$ can be an arbitrary ellipsotope). It is dual to the first approach in that it does not sample the vertices of $\breve{Z}$, but rather the facets of $\widehat{Z}$. Indeed, if $\widehat{Z}$ is non-degenerate and $Z(\underline{\boldsymbol{H}})$ is in halfspace representation as described in [10, Theorem 2.1] with halfspace matrix $\underline{\boldsymbol{\Lambda}}$ and coefficient vector $\vec{b}$ (note that none of the b_i can be 0, since $Z(\underline{\boldsymbol{H}})$ is non-degenerate), by using [4, Corollary 2] we can deduce

$$\begin{aligned} r(\breve{E}, \widehat{Z}) &= \max_{\|\vec{\alpha}\|_p \leq 1} \max_i \frac{|\vec{\lambda}_i^\top (\underline{\boldsymbol{G}}\vec{\alpha} + \vec{c} - \vec{d})|}{|b_i|} \\ &= \max_i \frac{\left\| \underline{\boldsymbol{G}}'^\top \vec{\lambda}_i \right\|_{p^*}}{|b_i|}, \end{aligned} \tag{11}$$

where $\vec{\lambda}_i$ is the i-th row of $\underline{\boldsymbol{\Lambda}}$, $\underline{\boldsymbol{G}}' = \begin{bmatrix} \underline{\boldsymbol{G}} & \vec{c} - \vec{d} \end{bmatrix}$, and the last equality follows by duality. The strategy is therefore to randomly select $n - 1$ out of the l generators from $Z(\underline{\boldsymbol{H}})$ and to compute the corresponding halfspace normal vector $\vec{\mathfrak{l}}$ and coefficient $\mathfrak{b}$ as in [10, Theorem 2.1] (note that we do **not** need to compute all halfspaces, which would lead to an exponential runtime; instead, we compute **only one** of the halfspace normal vectors, which has runtime $\mathcal{O}(n^4)$). Doing this N times yields N normal vectors $\vec{\mathfrak{l}}_1, ..., \vec{\mathfrak{l}}_N$ and coefficients $\mathfrak{b}_1, ..., \mathfrak{b}_N$ that lead to the approximation

$$\mathfrak{r}_{\text{hs}} := \max_{1 \leq i \leq N} \frac{\left\| \underline{\boldsymbol{G}}'^\top \vec{\mathfrak{l}}_i \right\|_{p^*}}{|\mathfrak{b}_i|}, \tag{12}$$

which satisfies $\mathfrak{r}_{\text{hs}} \leq r(\breve{Z}, \widehat{Z})$. We will call this approach *naive halfspace sampling*.

B. Shenmaier Halfspace Sampling

The two naive approaches can be evaluated quickly. However, our analysis so far offers no guarantees about the accuracy of the result, only that it under-approximates the true value $r(\breve{Z}, \widehat{Z})$. A result from [11] will allow us to devise algorithms for which we can estimate the likelihood that we are within a certain accuracy ϵ of the exact solution, without the assumption that $\breve{E}$ or $\widehat{E}$ should be a zonotope. We first need a technical definition:

Definition 4.1: Let $K \subset \mathbb{R}^n$ be compact with $\text{vol}(K) > 0$ and let $\vec{\mathfrak{v}} \in K$ be a random variable. Then $\vec{\mathfrak{v}}$ is said to be ϑ-uniformly distributed on K for some $\vartheta \geq 0$, if for any Lebesgue-measurable subset $S \subseteq K$

$$\left| \mathbb{P}(\vec{\mathfrak{v}} \in S) - \frac{\text{vol}(S)}{\text{vol}(K)} \right| \leq \vartheta. \tag{13}$$

Unless explicitly stated otherwise, for the present subsection and the next one, we assume $\vartheta > 0$ to be arbitrary but fixed.

Our main tool will be the following generalization of [11, Theorem 3]:

Theorem 4.2: Let $K \subset \mathbb{R}^n$ be compact with $\mathrm{vol}(K) > 0$, $\vec{c} \in \mathbb{R}^n$ a vector, $\|\cdot\|$ some norm on $\mathbb{R}^n$, and $\mathcal{B}$ the unit ball of the norm $\|\cdot\|$. For $N \in \mathbb{N}$, let $\vec{\mathfrak{v}}_i \in \mathcal{B}$ be independently, ϑ-uniformly sampled on $\mathcal{B}$, for $i = 1, ..., N$, and let

$$\vec{\mathfrak{z}}_i = \operatorname*{argmax}_{\vec{z} \in \mathcal{B}^*} \vec{z}^\top \vec{\mathfrak{v}}_i. \tag{14}$$

We define the approximation

$$\mathfrak{r}_{\mathrm{S}} := \max_i \max_{\vec{x} \in K} \vec{\mathfrak{z}}_i^\top \vec{x}. \tag{15}$$

Then

$$\mathfrak{r}_{\mathrm{S}} \leq \max_{\vec{x} \in K} \|\vec{x}\| \tag{16}$$

always holds, and for any $\varepsilon \in (0, 1]$ we have

$$(1 - \varepsilon) \cdot \max_{\vec{x} \in K} \|\vec{x}\| \leq \mathfrak{r}_{\mathrm{S}} \tag{17}$$

with a probability of at least

$$P_{\mathrm{S}}(\varepsilon) := 1 - \left(1 - \left(\frac{\varepsilon}{2 + \varepsilon}\right)^n + \vartheta\right)^N. \tag{18}$$

Note that $P_{\mathrm{S}}(\varepsilon)$ may become negative, in which case we treat it as $P_{\mathrm{S}}(\varepsilon) = 0$. For a proof of Theorem 4.2, see the appendix. If K is convex, the expression

$$\max_{\vec{x} \in K} \vec{\mathfrak{z}}_i^\top \vec{x}$$

coincides with the so-called *support function* of K evaluated at $\vec{\mathfrak{z}}_i$, which can usually be computed in polynomial time (it is the maximum of a linear function over a convex set), and even has a simple closed form for ellipsotopes: Given an ellipsotope $E = E_p(\vec{c}, \underline{\boldsymbol{G}}) = E_p(\underline{\boldsymbol{G}}) + \vec{c}$, using duality its support function in direction $\vec{z}$ can be computed as

$$\max_{\vec{x} \in E} \vec{z}^\top \vec{x} = \left\|\underline{\boldsymbol{G}}^\top \vec{z}\right\|_{p^*} + \vec{z}^\top \vec{c}. \tag{19}$$

Applying Theorem 4.2 to approximate (6), we end up with Algorithm 2, which provides the following result:

Theorem 4.3: Let $\breve{E}$, $\widehat{E}$, ϑ, $\mathfrak{r}_{\text{Shs}}$, and P_{Shs} be as in Algorithm 2, and assume that $\widehat{E}$ is non-degenerate. Then $\breve{E} \not\subseteq \widehat{E}$ if $\mathfrak{r}_{\text{Shs}} > 1$, and P_{Shs} is a lower bound on the probability that $\breve{E} \subseteq \widehat{E}$ if $\mathfrak{r}_{\text{Shs}} \leq 1$.

Proof: We can reformulate (6) to obtain

$$r(\breve{E}, \widehat{E}) = \max_{\vec{x} \in E_p(\underline{\boldsymbol{G}}) + \vec{c} - \vec{d}} \|\vec{x}\|_{E_q(\underline{\boldsymbol{H}})}.$$

On the other hand, in Algorithm 2 we define

$$\mathfrak{r}_{\text{Shs},i} \leftarrow \left\|\underline{\boldsymbol{G}}^\top \vec{\mathfrak{z}}_i\right\|_{p^*} + \vec{\mathfrak{z}}_i^\top (\vec{c} - \vec{d}). \tag{20}$$

As discussed in (19), the right-hand side of (20) coincides with the support function of $E_p(\underline{\boldsymbol{G}}) + \vec{c} - \vec{d} = E_p(\underline{\boldsymbol{G}}, \vec{c} - \vec{d})$ evaluated at the point $\vec{\mathfrak{z}}_i$, which entails

$$\mathfrak{r}_{\text{Shs},i} = \max_{\vec{x} \in E_p(\underline{\boldsymbol{G}}) + \vec{c} - \vec{d}} \vec{\mathfrak{z}}_i^\top \vec{x}.$$

By Theorem 4.2, we can conclude $r(\breve{E}, \widehat{E}) \geq \mathfrak{r}_{\text{Shs}}$, and additionally $(1 - \varepsilon) r(\breve{E}, \widehat{E}) \leq \mathfrak{r}_{\text{Shs}}$ holds with probability $P_{\mathrm{S}}(\varepsilon)$ for any $\varepsilon \in (0, 1]$. So, if $\mathfrak{r}_{\text{Shs}} > 1$, it follows that $r(\breve{E}, \widehat{E}) > 1$, which is equivalent to $\breve{E} \not\subseteq \widehat{E}$. If, on the other hand, $\mathfrak{r}_{\text{Shs}} \leq 1$, we may choose $\varepsilon = 1 - \mathfrak{r}_{\text{Shs}}$, and conclude that $r(\breve{E}, \widehat{E}) \leq 1$ holds with a probability of at least $P_{\mathrm{S}}(1 - \mathfrak{r}_{\text{Shs}}) = P_{\text{Shs}}$, which implies $\breve{E} \subseteq \widehat{E}$ with the same probability. ∎

Remark 4.4: Line 2 of Algorithm 2 requires solving a nonlinear optimization problem, which is difficult in general. However, since the constraint is convex and the objective function is linear, this can be solved efficiently in polynomial time, for example, using interior point algorithms (see [8, Chapter 11]).

Remark 4.5: To apply Algorithm 2, it remains to discuss how to uniformly sample points on an ellipsotope $E_q(\underline{\boldsymbol{H}})$. As suggested in [11], this can be done using the ball walk algorithm described in [12, Section 3b.]. The conditions for applying the ball walk are addressed in the appendix.

Algorithm 2 Shenmaier Halfspace Sampling

Input: Non-degenerate ellipsotopes $\breve{E} = E_p(\underline{\boldsymbol{G}}, \vec{c})$ and $\widehat{E} = E_q(\underline{\boldsymbol{H}}, \vec{d})$ in $\mathbb{R}^n$, number of samples N, parameter $\vartheta > 0$.
Output: False if $\breve{E} \not\subseteq \widehat{E}$ could be confirmed, otherwise output a lower bound P_{Shs} on the probability that $\breve{E} \subseteq \widehat{E}$.

1: Draw $\vec{\mathfrak{v}}_1, \cdots, \vec{\mathfrak{v}}_N$ independently, ϑ-uniformly from $E_q(\underline{\boldsymbol{H}})$
2: For $i = 1, \cdots, N$, compute

$$\vec{\mathfrak{z}}_i \leftarrow \operatorname*{argmax}_{\left\|\underline{\boldsymbol{H}}^\top \vec{z}\right\|_q^* \leq 1} \vec{z}^\top \vec{\mathfrak{v}}_i$$

3: $\mathfrak{r}_{\text{Shs},i,\vartheta} \leftarrow \left\|\underline{\boldsymbol{G}}^\top \vec{\mathfrak{z}}_i\right\|_{p^*} + \vec{\mathfrak{z}}_i^\top (\vec{c} - \vec{d})$
4: $\mathfrak{r}_{\text{Shs}} \leftarrow \max_i \mathfrak{r}_{\text{Shs},i}$
5: **if** $\mathfrak{r}_{\text{Shs}} > 1$ **then**
6: **return** False
7: **else**
8: $P_{\text{Shs}} \leftarrow P_{\mathrm{S}}(1 - \mathfrak{r}_{\text{Shs}})$
9: **return** P_{Shs}
10: **end if**

C. Shenmaier Vertex Sampling

In Lines 1 and 2 of Algorithm 2, we essentially sample points from the boundary of $(E(\underline{\boldsymbol{H}})_q)^*$, which by duality correspond to halfspaces containing $E(\underline{\boldsymbol{H}})_q$. We then compute the size of $\breve{E} - \vec{d}$ in the direction of those halfspaces, which is reminiscent of the naive halfspace sampling algorithm. Therefore, one may ask oneself whether another method could correspond to the naive vertex sampling. This is indeed the case if we apply Theorem 4.2 to (7) instead of (6). This time, instead of sampling points on the boundary of $(E(\underline{\boldsymbol{H}})_q)^*$, we sample points on the boundary of $\left\{\vec{x} \in \mathbb{R}^n \;\middle|\; \left\|\underline{\boldsymbol{G}}^\top \vec{x}\right\|_{p^*} \leq 1\right\}^* = E_p(\underline{\boldsymbol{G}})$, which is similar to the naive vertex sampling. Furthermore, instead of sampling points from $E_p(\underline{\boldsymbol{G}})$, we can use Theorem 4.2 on $K = \underline{\boldsymbol{G}}^\top (E_q(\underline{\boldsymbol{H}}))^*$, which means that we only need to sample points on $\mathcal{B}_{p^*}$, which is far easier (see [13, Theorem 1]). The overall method is depicted in Algorithm 3, where the notation $|[\vec{x}]|^s$ refers to the vector with coordinates $|x_i|^s$.

Theorem 4.6: Let $\breve{E}$, $\widehat{E}$, $\mathfrak{r}_{\text{Svs}}$, and P_{Svs} be as in Algorithm 3. Then $\breve{E} \not\subseteq \widehat{E}$ if $\mathfrak{r}_{\text{Svs}} > 1$, and P_{Svs} is a lower bound on the probability that $\breve{E} \subseteq \widehat{E}$ if $\mathfrak{r}_{\text{Svs}} \leq 1$.

Proof: The proof is essentially the same as that of Theorem 4.3, using the fact that, for any $\vec{x} \in \mathbb{R}^n$,

$$\underset{\vec{z}\in\mathcal{B}_p}{\operatorname{argmax}} \vec{z}^\top \vec{x} = \frac{\operatorname{sign}(\vec{x}) \circ |[\vec{x}]|^{p^*-1}}{\|\vec{x}\|_{p^*}^{p^*-1}}, \tag{21}$$

as mentioned, e.g., in [14, Section 2.2]. ∎

D. Revisiting the Naive Algorithms

Using similar techniques as for the proofs of Theorem 4.3 and Theorem 4.6, we can improve our accuracy assessment of the naive algorithms presented in Section IV-A. For the remainder of this section, we take $\vartheta = 0$ since Section IV-A only involves uniform sampling over certain domains.

Corollary 4.7: For a zonotope $\breve{Z}$ and an ellipsotope $\widehat{E}$ in $\mathbb{R}^n$, let $\mathfrak{r}_{\text{vs}}$ be the result of the naive vertex sampling algorithm as described in Section IV-A.1. Then $\breve{Z} \not\subseteq \widehat{E}$ if $\mathfrak{r}_{\text{vs}} > 1$, and $\breve{Z} \subseteq \widehat{E}$ with a probability of at least $P_{\text{S}}(1-\mathfrak{r}_{\text{vs}})$ if $\mathfrak{r}_{\text{vs}} \leq 1$.

Proof: This follows directly from Theorem 4.6, since Algorithm 3 reduces to the naive vertex sampling algorithm if $\breve{E}$ is a zonotope. ∎

The corresponding corollary for the halfspace sampling can not be deduced easily from Theorem 4.3, so we need another argument:

Corollary 4.8: For an ellipsotope $\breve{E}$ and a non-degenerate zonotope $\widehat{Z}$ in $\mathbb{R}^n$, let $\mathfrak{r}_{\text{hs}}$ be the result of the naive halfspace sampling algorithm as described in Section IV-A.2. Then $\breve{E} \not\subseteq \widehat{Z}$ if $\mathfrak{r}_{\text{hs}} > 1$, and $\breve{E} \subseteq \widehat{Z}$ with a probability of at least $P_{\text{S}}(1-\mathfrak{r}_{\text{hs}})$ if $\mathfrak{r}_{\text{hs}} \leq 1$.

Proof: Suppose $\breve{E}$ and $\widehat{Z}$ are given as $\breve{E} = E_p(\underline{\boldsymbol{G}}, \vec{c})$ and $\widehat{Z} = Z(\underline{\boldsymbol{H}}, \vec{d})$, respectively, let $\underline{\boldsymbol{\Lambda}}$ and $\vec{b}$ be a halfspace representation of $Z(\underline{\boldsymbol{H}})$, with $\vec{\lambda}_j$ being the rows of $\underline{\boldsymbol{\Lambda}}$ as described in Section IV-A.2, and let k be the number of such halfspaces. It follows from (11) that

$$r(\breve{E}, \widehat{Z}) = \max_j \frac{\left\| \underline{\boldsymbol{G}}'^\top \vec{\lambda}_j \right\|_{p^*}}{|b_j|},$$

Algorithm 3 Shenmaier Vertex Sampling

Input: Non-degenerate ellipsotopes $\breve{E} = E_p(\underline{\boldsymbol{G}}, \vec{c})$ and $\widehat{E} = E_q(\underline{\boldsymbol{H}}, \vec{d})$ in $\mathbb{R}^n$, number of samples N.
Output: False if $\breve{E} \not\subseteq \widehat{E}$ could be confirmed, otherwise output a lower bound P_{Svs} on the probability that $\breve{E} \subseteq \widehat{E}$.

1: Draw $\vec{\mathfrak{v}}_1, \cdots, \vec{\mathfrak{v}}_N$ uniformly from $\mathcal{B}_{p^*}$ using [13, Theorem 1]
2: For $i = 1, \cdots, N$, compute

$$\vec{\mathfrak{z}}_i \leftarrow \frac{\operatorname{sign}(\vec{\mathfrak{v}}_i) \circ |[\vec{\mathfrak{v}}_i]|^{p^*-1})}{\left\|\vec{\mathfrak{v}}_i\right\|_{p^*}^{p^*-1}}$$

3: $\mathfrak{r}_{\text{Svs},i} \leftarrow \left\| \underline{\boldsymbol{G}} \vec{\mathfrak{z}}_i + \vec{c} - \vec{d} \right\|_{E_q(\underline{\boldsymbol{H}})}$
4: $\mathfrak{r}_{\text{Svs}} \leftarrow \max_i \mathfrak{r}_{\text{Svs},i}$
5: **if** $\mathfrak{r}_{\text{Svs}} > 1$ **then**
6: **return** False
7: **else**
8: $P_{\text{Svs}} \leftarrow P_{\text{S}}(1 - \mathfrak{r}_{\text{Svs}})$, with ϑ set to 0
9: **return** P_{Svs}
10: **end if**

with $\underline{\boldsymbol{G}}' = \begin{bmatrix} \underline{\boldsymbol{G}} & \vec{c} - \vec{d} \end{bmatrix}$. Using the Bauer maximum principle, it is easy to see that this can be reformulated as

$$r(\breve{E}, \widehat{Z}) = \max_{\|\vec{x}\|_1 \leq 1} \left\| \underline{\boldsymbol{G}}'^\top \underline{\boldsymbol{M}} \vec{x} \right\|_{p^*},$$

where $\underline{\boldsymbol{M}}$ is the matrix whose columns are the vectors $\vec{\lambda}_j / |b_j|$. Using duality, one can show

$$r(\breve{E}, \widehat{Z}) = \max_{\|\vec{y}\|_p \leq 1} \left\| \underline{\boldsymbol{M}}^\top \underline{\boldsymbol{G}}' \vec{y} \right\|_\infty. \tag{22}$$

Suppose $\vec{\mathfrak{v}}$ is chosen uniformly on $\mathcal{B}_\infty$. Then, it is easy to see that the vector

$$\vec{\mathfrak{z}} = \underset{\|\vec{z}\|_1 \leq 1}{\operatorname{argmax}} \vec{z}^\top \vec{\mathfrak{v}}$$

can be assumed to be one of the vertices of $\mathcal{B}_1$ by using the Bauer maximum principle. Since these vertices are just the vectors $\pm\vec{e}_s$, where $s = 1, ..., k$, we may assume that $\vec{\mathfrak{z}}$ equals one of these vectors with equal probability, which in turn means that $\underline{\boldsymbol{M}} \vec{\mathfrak{z}} = \pm \vec{\lambda}_j / |b_j|$ for some $i = j, ..., k$, each with equal probability. Therefore, $\underline{\boldsymbol{M}} \vec{\mathfrak{z}}$ randomly selects a halfspace vector $\vec{\mathfrak{l}}$ and a coefficient $\mathfrak{b}$ from $\vec{\lambda}_j$, b_j, just like for the naive halfspace sampling algorithm. It is then easy to see that the approximation given by Theorem 4.2 applied on (22) yields the same result as the naive halfspace sampling procedure. The rest of the Corollary can then be proven similarly to the proof of Theorem 4.3. ∎

V. Numerical Results

We demonstrate the efficiency of our novel algorithms for solving the containment problem by modeling zonotopes and ellipsoids (i.e., 2-ellipsotopes) using the CORA toolbox [15]. We solve linear programs using the MOSEK optimization suite [16]. All computations were made in MATLAB on an Intel Core i7-8650U CPU @1.9GHz with 24GB memory.[1]

A. Advantages of Tree Search

For our first experiment, we confirm the efficiency of the search-based method described in Algorithm 1 compared to the original, brute-force vertex enumeration from [4, Algorithm 1]. We evaluated both algorithms on zonotopes $\breve{Z}$ and $\widehat{Z}$ in $\mathbb{R}^5$ with 10 generators, both centered at the origin with generator entries randomly and uniformly sampled within $[-1, 1]$. Additionally, to better analyze in which situations each algorithm performs best, we multiplied all entries of the generators of $\breve{Z}$ with a factor $\varrho \in [0, 1.2]$, which we changed over different runs. We did so for 100 zonotope pairs for each run and present the results in Fig. 1, where the points represent the average over 100 zonotope pairs and the error bars correspond to the worst case and best case runtimes (not the standard deviation). Our new search-based approach yields significantly better results for smaller ϱ, which corresponds to the case where $\breve{Z}$ is much smaller on average than $\widehat{Z}$, as the original vertex enumeration method needs to cycle through every single vertex to confirm that

[1]The scripts of our results, including the code generating the figures in this document, are available at the URL `https://github.com/AdrianKulmburg/ellipsotopeContainmentProblem`

it is indeed contained in $\widehat{Z}$, whereas Algorithm 1 can make use of the heuristic to quickly determine that no vertex can possibly be outside of $\widehat{Z}$. On the other hand, for higher values of ϱ, the likelihood that $\check{Z}$ is not contained in $\widehat{Z}$ is high enough that pretty much any vertex of $\check{Z}$ is not contained in $\widehat{Z}$, so the common vertex enumeration technique is likely to quickly find a vertex that lies outside of $\widehat{Z}$. The same holds for the search-based method, even though it requires a bit more time because it needs to sort the length of the generators beforehand. Overall, this additional runtime is relatively insignificant. We also compared our runtimes to those of the approximative algorithm given in [5, Corollary 4], but did not include the results in Fig. 1 for the sake of readability; in all instances, the approximative algorithm was faster than the search-based approach (on average by a factor of 0.5). However, as mentioned in the introduction, this should be taken with a grain of salt, since the approach from [5, Corollary 4] is not exact, whereas our search-based approach is.

B. Accuracy of the Sampling Methods

We now compare the efficiency of the vertex and halfspace sampling algorithms for the zonotope containment problem. We computed $\mathfrak{r}_{\text{vs}}$ and $\mathfrak{r}_{\text{hs}}$ for different values of N and compared the results to the exact value of $r(\check{Z}, \widehat{Z})$ computed using the search-based vertex enumeration. Specifically, we calculated the approximation ratio

$$\rho = \frac{\mathfrak{r}}{r(\check{Z}, \widehat{Z})}, \tag{23}$$

where $\mathfrak{r}$ is either $\mathfrak{r}_{\text{vs}}$ or $\mathfrak{r}_{\text{hs}}$. For each value of N, we computed the average over 100 zonotope pairs that were chosen just like in the previous experiment. The results are shown in Fig. 2 (just like for Fig. 1, the points correspond to the average over 100 runs, while the errorbars correspond to the best and worst cases). To compare those results to the accuracy predicted by Theorem 4.2, we added the value $1 - \varepsilon_{99.99\%}$ that corresponds to the case where $P_{\text{S}}(\varepsilon_{99.99\%}) = 99.99\%$ (when the value $1 - \varepsilon_{99.99\%}$ was negative for some N, we interpreted it as being equal to 0 instead). We also summarized the average runtime (over the 100 zonotope pairs) of each algorithm in the case of $N = 10^4$ samples in Table I.

TABLE I

AVERAGE RUNTIMES OF EACH ALGORITHM, FOR $N = 10^4$ SAMPLES. ALL VALUES ARE DISPLAYED IN MILLISECONDS.

Algorithm	Exact Search (Algorithm 1)	Vertex Sampling	Halfspace Sampling
Runtime	466.6±24.5	3177.5±29.2	93.1±0.2

As one can see in Fig. 2, the overall accuracy of both sampling algorithms is always better than that predicted by Theorem 4.2 with probability 99.99%, with the halfspace sampling typically performing better than the vertex sampling. While this shows that Theorem 4.2 does indeed yield a plausible lower bound for the accuracy, it also indicates that this bound is perhaps too conservative, so that tighter accuracy estimations could be deduced in the future. On the other hand, Table I shows that halfspace sampling has a much lower runtime than vertex sampling, the latter one performing even worse than the exact calculation using Algorithm 1. While this tendency could be reversed for cases where $\check{Z}$ has a significantly lower number of generators than $\widehat{Z}$, in the general case the halfspace sampling algorithm seems to be the better choice in terms of accuracy and runtime.

C. Quadrotor Reaching a Target Area

To illustrate how solving the containment problem for ellipsotopes can be useful in practice, we consider a quadrotor changing its altitude and moving to an ellipsoidal target area. We assume that the state of the quadrotor is characterized by the following variables: The inertial (north) position x_1, the

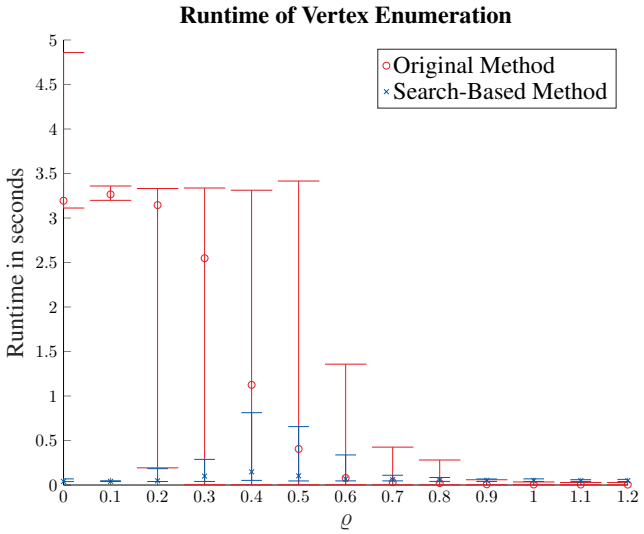

Fig. 1. Runtime of the original vertex enumeration algorithm, compared to that of the search-based algorithm, for varying sizes of the inner zonotope.

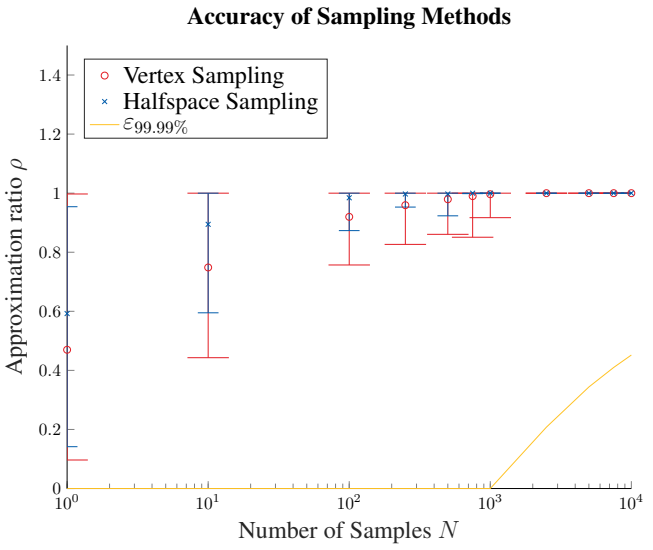

Fig. 2. Comparison of the accuracy of the sampling methods against the expected accuracy according to Theorem 4.2.

inertial (east) position x_2, the altitude x_3, the longitudinal velocity x_4, the lateral velocity x_5, the vertical velocity x_6, the roll angle x_7, the pitch angle x_8, the yaw angle x_9, the roll rate x_{10}, the pitch rate x_{11}, and the yaw rate x_{12}. The quadrotor obeys the dynamics derived in [17, Equations (16)-(19)] and similar to the specifications laid out in [18, Section 3.3.2], the initial position of the quadrotor is uncertain in all directions within $[-0.4, 0.4]$. Contrary to [18], we assume that the quadrotor is immobile at the start so that the velocity is zero in all directions. The remaining variables are also zero in the beginning. Finally, we use the same approach as in [18, Section 3.3.1] to control the behavior of the quadrotor, resulting in the input variables u_1, u_2, and u_3, which control the height, roll, and pitch, respectively. We use the constant input values $u_1 = 1$, $u_2 = 0$, and $u_3 = -0.05$. The target area for x_1, x_2, and x_3 is modeled as an ellipsoid $E_2(\underline{\boldsymbol{G}}, \vec{c})$, with $\underline{\boldsymbol{G}} = \operatorname{diag}(0.8, 1, 0.1)$ and $\vec{c} = (4, 0, 1)^\top$. For the containment check, we apply Algorithm 1 using the fact that $\|\vec{x}\|_{E_2(\underline{\boldsymbol{G}})} = \sqrt{\vec{x}^\top \underline{\boldsymbol{G}}^{-1\top} \underline{\boldsymbol{G}}^{-1} \vec{x}}$ for an invertible matrix $\underline{\boldsymbol{G}}$. The reachable sets are represented as zonotopes and computed using the CORA toolbox [15]. As seen in Fig. 3, the target area contains the final reachable set. The containment check took only 0.84 seconds despite the final set having 600 generators.

VI. Conclusion

We present a deterministic algorithm based on a binary tree search that performs better at solving exactly the zonotope containment problem in practice than the vertex enumeration in [4], even though it may still have exponential runtime in the worst case. In addition, we present two techniques to solve the ellipsotope containment problem via sampling. In practice, these algorithms are significantly more accurate than the theoretical lower bound we derived in Section IV, which indicates that this bound may be improved in the future. Overall, the new deterministic algorithm is particularly suitable for cases where containment has to be computed exactly, at the cost of an exponential runtime in the worst case. As for the sampling-based algorithms, they can ideally be used after the method in [5, Corollary 4] (which can quickly confirm containment in most cases) has been used without being able to confirm containment, since the sampling-based algorithms have a controllable accuracy and runtime.

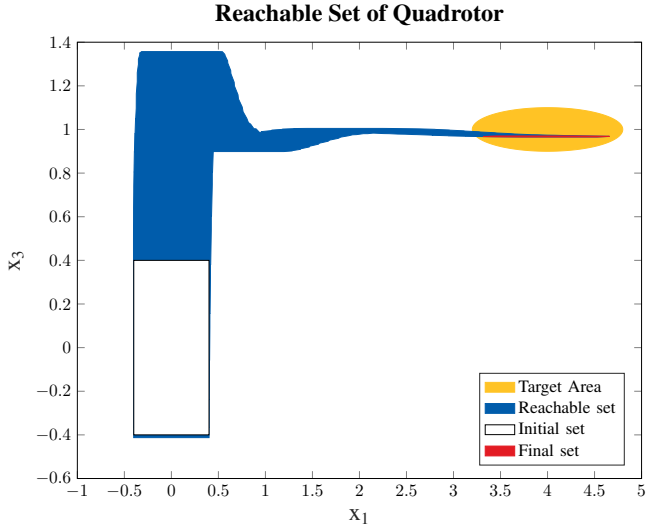

Fig. 3. Reachable set computation of a quadrotor. The quadrotor reaches the ellipsoidal target area, which can be confirmed using Algorithm 1.

Appendix

A. Proof of Theorem 4.2

In this section, we show how to prove Theorem 4.2. Since (16) is obvious, we will only show (17). We begin with a similar approach to that of the proof of [11, Theorem 3]: Let $\vec{x}^\triangle$ be a maximizer of

$$\max_{\vec{x} \in K} \|\vec{x}\| .$$

If $\vec{x} = \vec{0}$, the proof is trivial, so assume without loss of generality that $\vec{x} \neq \vec{0}$ and define $\vec{\chi}^\triangle = \vec{x}^\triangle / \left\|\vec{x}^\triangle\right\|$. By construction, the points $(1+\varepsilon/2)\vec{\mathfrak{v}}_i$ are ϑ-uniformly distributed in $(1+\varepsilon/2)\mathcal{B}$. Moreover, since $\vec{\chi}^\triangle$ lies on the boundary of $\mathcal{B}$, the set $\mathcal{B}_{\varepsilon/2}(\vec{\chi}^\triangle) := \vec{\chi}^\triangle + \frac{\varepsilon}{2} \cdot \mathcal{B}$ is contained in $(1+\varepsilon/2)\mathcal{B}$. This means that the probability of one of the points $(1+\varepsilon/2)\vec{\mathfrak{v}}_i$ hitting $\mathcal{B}_{\varepsilon/2}(\vec{\chi}^\triangle)$ is at least

$$\frac{\operatorname{vol}(\mathcal{B}_{\varepsilon/2}(\vec{\chi}^\triangle))}{\operatorname{vol}((1+\varepsilon/2)\mathcal{B})} - \vartheta = \frac{(\varepsilon/2)^n}{(1+\varepsilon/2)^n} - \vartheta = \left(\frac{\varepsilon}{2+\varepsilon}\right)^n - \vartheta.$$

Therefore, the probability that at least one of the N points $(1+\varepsilon/2)\vec{\mathfrak{v}}_i$ hits the ball $\mathcal{B}_{\varepsilon/2}(\vec{\chi}^\triangle)$ is $P_{\mathrm{S}}(\varepsilon) = 1 - \left(1 - \left(\frac{\varepsilon}{2+\varepsilon}\right)^n + \vartheta\right)^N$. Let $\vec{\mathfrak{v}}^\triangle$ be such that $(1+\varepsilon/2)\vec{\mathfrak{v}}^\triangle$ is the closest point to $\vec{\chi}^\triangle$ (with respect to the $\|\cdot\|$-norm). Then, by what we discussed above, there holds $\left\|\vec{\chi}^\triangle - (1+\varepsilon/2)\vec{\mathfrak{v}}^\triangle\right\| \leq \varepsilon/2$ with a probability of at least $P_{\mathrm{S}}(\varepsilon)$. Let $\vec{\mathfrak{w}}^\triangle = \vec{\mathfrak{v}}^\triangle / \left\|\vec{\mathfrak{v}}^\triangle\right\|$ if $\vec{\mathfrak{v}}^\triangle \neq \vec{0}$, and $\vec{\mathfrak{w}}^\triangle = \vec{0}$ otherwise. It follows by the triangle inequality that, if $\vec{\mathfrak{w}}^\triangle \neq \vec{0}$, with a probability of at least $P_{\mathrm{S}}(\varepsilon)$,

$$\begin{aligned}\left\|\vec{\chi}^\triangle - \vec{\mathfrak{w}}^\triangle\right\| &\leq \left\|\vec{\chi}^\triangle - (1+\frac{\varepsilon}{2})\vec{\mathfrak{v}}^\triangle\right\| + \left\|(1+\frac{\varepsilon}{2})\vec{\mathfrak{v}}^\triangle - \vec{\mathfrak{w}}^\triangle\right\| \\ &\leq \frac{\varepsilon}{2} + (1+\frac{\epsilon}{2} - \frac{1}{\vec{\mathfrak{v}}^\triangle})\left\|\vec{\mathfrak{v}}^\triangle\right\| \\ &\leq \varepsilon\end{aligned}$$

For the last inequality, we utilize the fact that $\vec{\mathfrak{v}}^\triangle \in \mathcal{B}$, and thus $\left\|\vec{\mathfrak{v}}^\triangle\right\| \leq 1$. Instead, if $\vec{\mathfrak{w}}^\triangle = \vec{0}$, it is easy to check that $\left\|\vec{\chi}^\triangle - \vec{\mathfrak{w}}^\triangle\right\| \leq \varepsilon/2 \leq \varepsilon$. Let $\vec{\mathfrak{z}}^\triangle$ be the vector in (14) corresponding to $\vec{\mathfrak{v}}^\triangle$. Note that, by definition and using duality, it follows that $\vec{\mathfrak{z}}^{\triangle\top}\vec{\mathfrak{v}}^\triangle = \left\|\vec{\mathfrak{v}}^\triangle\right\|$. Then,

$$\begin{aligned}\max_i \vec{\mathfrak{z}}_i^\top \vec{\chi}^\triangle &= \max_i \vec{\mathfrak{z}}_i^\top (\vec{\mathfrak{w}}^\triangle - (\vec{\mathfrak{w}}^\triangle - \vec{\chi}^\triangle)) \\ &\geq \left(\max_i \vec{\mathfrak{z}}_i^\top \vec{\mathfrak{w}}^\triangle\right) - \left(\max_i \vec{\mathfrak{z}}_i^\top (\vec{\mathfrak{w}}^\triangle - \vec{\chi}^\triangle)\right) \\ &\geq \vec{\mathfrak{z}}^{\triangle\top}\vec{\mathfrak{w}}^\triangle - \max_{\vec{z} \in \mathcal{B}^*} \vec{z}^\top (\vec{\mathfrak{w}}^\triangle - \vec{\chi}^\triangle) \\ &= 1 - \left\|\vec{\mathfrak{w}}^\triangle - \vec{\chi}^\triangle\right\| \\ &= 1 - \varepsilon\end{aligned}$$

where for the first inequality, we have used the fact that, for any two functions $f(x)$ and $g(x)$, we have $\max_x(f(x) - g(x)) \geq \max_x f(x) - \max_x g(x)$ (this can be proven just like the inverse triangle inequality, using the fact that for any two functions $f(x)$ and $g(x)$, we have $\max_x(f(x)+g(x)) \leq \max_x f(x) + \max_x g(x)$). We can thus conclude that, with a probability of at least $P_{\mathrm{S}}(\varepsilon)$,

$$\begin{aligned}\max_i \max_{\vec{x}\in K} \vec{\mathfrak{d}}_i^\top \vec{x} &= \max_{\vec{x}\in K} \max_i \vec{\mathfrak{d}}_i^\top \vec{x} \\ &\geq \max_i \vec{\mathfrak{d}}_i^\top \vec{x}^\triangle \\ &= \max_i (\vec{\mathfrak{d}}_i^\top \vec{\chi}^\triangle) \left\|\vec{x}^\triangle\right\| \\ &\geq (1-\varepsilon) \left\|\vec{x}^\triangle\right\| \\ &= (1-\varepsilon) \max_{\vec{x}\in K} \|\vec{x}\|\end{aligned}$$

□

B. Uniform Sampling within Ellipsotopes

We now discuss the pre-processing steps that are required to apply the ball walk algorithm from [12, Section 3b.] to ellipsotopes. The ball walk yields a point that is ϑ-uniformly distributed, provided that the convex, compact set S on which it is applied satisfies the following two conditions:

1) More than 2/3 of the volume of the unit ball $\mathcal{B}_2$ is contained in S.
2) For some parameter $1 \leq \xi \leq n^{3/2}$, more than 2/3 of the volume of S is contained in $\xi\mathcal{B}_2$.

One common technique to ensure that these two conditions are met is called *rounding*, and consists in transforming S beforehand in such a way that it matches as closely as possible the unit ball. For ellipsotopes, this is particularly easy: Suppose $q \in [2, \infty]$, then there holds

$$\mathcal{B}_2 \subseteq \mathcal{B}_q \subseteq n^{1/2-1/q}\mathcal{B}_2.$$

The ellipsotope $E_q(\underline{\boldsymbol{H}})$ can equivalently be described as $E_q(\underline{\boldsymbol{H}}) = \underline{\boldsymbol{H}}\mathcal{B}_q$, therefore we may write

$$\underline{\boldsymbol{H}}\mathcal{B}_2 \subseteq E_q(\underline{\boldsymbol{H}}) \subseteq n^{1/2-1/q}\underline{\boldsymbol{H}}\mathcal{B}_2.$$

Let $\underline{\boldsymbol{H}} = \underline{\boldsymbol{U}}\,\underline{\boldsymbol{\Sigma}}\,\underline{\boldsymbol{V}}^\top$ be the singular value decomposition of $\underline{\boldsymbol{H}}$. Since $\underline{\boldsymbol{U}}$ and $\underline{\boldsymbol{V}}$ are, by construction, orthogonal matrices, they are both invertible and leave the unit ball $\mathcal{B}_2$ invariant, i.e., $\underline{\boldsymbol{U}}\mathcal{B}_2 = \mathcal{B}_2$ and $\underline{\boldsymbol{V}}\mathcal{B}_2 = \mathcal{B}_2$. Furthermore, since we may assume that $\underline{\boldsymbol{H}}$ is full-rank (since $E_q(\underline{\boldsymbol{H}})$ is non-degenerate), it follows that $\underline{\boldsymbol{\Sigma}}$ has the form $\underline{\boldsymbol{\Sigma}} = \begin{bmatrix}\underline{\boldsymbol{D}} & \underline{\mathbf{0}}\end{bmatrix}$ for some invertible, diagonal matrix $\underline{\boldsymbol{D}}$. This allows us to conclude that

$$\mathcal{B}_2 \subseteq E_q(\underline{\boldsymbol{D}}^{-1}\underline{\boldsymbol{U}}^\top\underline{\boldsymbol{H}}) \subseteq n^{1/2-1/q}\mathcal{B}_2.$$

For $q \in [1, 2]$, the analysis is similar, except that we end up with

$$\mathcal{B}_2 \subseteq n^{1/q-1/2}E_q(\underline{\boldsymbol{D}}^{-1}\underline{\boldsymbol{U}}^\top\underline{\boldsymbol{H}}) \subseteq n^{1/q-1/2}\mathcal{B}_2.$$

Therefore, in order to sample points on $E_q(\underline{\boldsymbol{H}})$, one can apply the ball-walk algorithm on the set $S = E_q(\underline{\boldsymbol{D}}^{-1}\underline{\boldsymbol{U}}^\top\underline{\boldsymbol{H}})$ if $q \in [2, \infty]$ (or $S = n^{1/q-1/2}E_q(\underline{\boldsymbol{D}}^{-1}\underline{\boldsymbol{U}}^\top\underline{\boldsymbol{H}})$ if $q \in [1, 2]$). We multiply the resulting points by $\underline{\boldsymbol{U}}\,\underline{\boldsymbol{D}}$ (or $n^{1/2-1/q}\underline{\boldsymbol{U}}\,\underline{\boldsymbol{D}}$) to obtain ϑ-uniformly distributed points on $E_q(\underline{\boldsymbol{H}})$.

This entire procedure can be performed in polynomial time with respect to the dimension n and the number of generators m of $E_q(\underline{\boldsymbol{H}})$, since the runtime of computing the singular value decomposition is well-known to be polynomial with respect to the size of the matrix $\underline{\boldsymbol{H}}$, and the ball walk algorithm has polynomial runtime according to [12, Theorem 3.7.].

References

[1] S. V. Raković, E. C. Kerrigan, D. Q. Mayne, and K. I. Kouramas, "Optimized robust control invariance for linear discrete-time systems: Theoretical foundations," *Automatica*, vol. 43, pp. 831–841, 2007.

[2] W. Ren, J. Calbert, and R. Jungers, "Zonotope-based Controller Synthesis for LTL Specifications," in *Proc. of the 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 580–585.

[3] H. Roehm, J. Oehlerking, M. Woehrle, and M. Althoff, "Reachset Conformance Testing of Hybrid Automata," in *Proc. of Hybrid Systems: Computation and Control*, 2016, pp. 277–286.

[4] A. Kulmburg and M. Althoff, "On the co-NP-completeness of the zonotope containment problem," *European Journal of Control*, vol. 62, pp. 84–91, 2021.

[5] S. Sadraddini and R. Tedrake, "Linear Encodings for Polytope Containment Problems," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 4367–4372.

[6] S. Kousik, A. Dai, and G. X. Gao, "Ellipsotopes: Uniting Ellipsoids and Zonotopes for Reachability Analysis and Fault Detection," *IEEE Transactions on Automatic Control*, pp. 1–13, 2022.

[7] T. Bühler and D. Salamon, *Functional Analysis*, ser. Graduate studies in mathematics. Providence, Rhodes Island: American Mathematical Society, 2018.

[8] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK ; New York: Cambridge University Press, 2004.

[9] A. Kulmburg, "The Generalized Matrix Norm Problem," 2023, to appear in the SIAM Journal on Matrix Analysis and Applications (SIMAX).

[10] M. Althoff, "Reachability Analysis and its Application to the Safety Assessment of Autonomous Cars," Ph.D. dissertation, Technical University of Munich, 2010.

[11] V. V. Shenmaier, "Complexity and Approximation of Finding the Longest Vector Sum," *Computational Mathematics and Mathematical Physics*, vol. 58, no. 6, pp. 850–857, 2018.

[12] L. Lovász and M. Simonovits, "Random walks in a convex body and an improved volume algorithm," *Random Structures and Algorithms*, vol. 4, pp. 359–412, 1993.

[13] F. Barthe, O. Guédon, S. Mendelson, and A. Naor, "A probabilistic approach to the geometry of the ℓ_p^n-ball," *The Annals of Probability*, vol. 33, no. 2, pp. 480 – 513, 2005.

[14] V. Bhattiprolu, M. Ghosh, V. Guruswami, E. Lee, and M. Tulsiani, "Approximating Operator Norms via Generalized Krivine Rounding," Electronic Colloquium on Computational Complexity, Tech. Rep. 097, 2018.

[15] M. Althoff, "An introduction to CORA 2015," in *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, vol. 34, 2015, pp. 120 – 151.

[16] M. ApS, *The MOSEK optimization toolbox for MATLAB manual. Version 10.0.*, 2022. [Online]. Available: http://docs.mosek.com/9.0/toolbox/index.html

[17] R. W. Beard, "Quadrotor Dynamics and Control Rev 0.1, Technical Report," 2008.

[18] F. Immler, M. Althoff, X. Chen, C. Fan, G. Frehse, N. Kochdumper, Y. Li, S. Mitra, M. S. Tomar, and M. Zamani, "ARCH-COMP18 Category Report: Continuous and Hybrid Systems with Nonlinear Dynamics," in *ARCH18. 5th International Workshop on Applied Verification of Continuous and Hybrid Systems*, vol. 54. EasyChair, 2018, pp. 53–70.