## SUPPLEMENT A
## POLYNOMIAL ZONOTOPE ARITHMETIC

We introduce several useful operations on polynomial zonotopes: interval conversions, set addition and multiplication, slicing, computing bounds, and set-valued function evaluation.

*1) Interval Arithmetic:* Let $\mathbb{IR}^n$ be the set of all real-valued $n$-dimensional interval vectors. The Minkowski sum and difference of $[x]$ and $[y]$ are

$$[x] \oplus [y] = [\underline{x}+\underline{y}, \overline{x}+\overline{y}], \tag{S.1}$$

$$[x] \ominus [y] = [\underline{x}-\overline{y}, \overline{x}-\underline{y}]. \tag{S.2}$$

The product of $[x]$ and $[y]$ is

$$[x][y] = \left[\min\left(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\right), \max\left(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\right)\right]. \tag{S.3}$$

The $i^{\text{th}}$ and $j^{\text{th}}$ entry of the product of an interval matrix $[Y]$ multplied by an interval matrix $[X]$ is

$$([X][Y])_{ij} = \bigoplus_{k=1}^{n} ([X]_{ik}[Y]_{kj}), \tag{S.4}$$

where $n$ is the number of columns of $[X]$ and number of rows of $[Y]$. Given interval vectors $[x], [y] \subset \mathbb{R}^3$, their cross product is

$$[x] \otimes [y] = [x]^{\times}[y], \tag{S.5}$$

where $[x]^{\times}$ is the skew-symmetric matrix representation of $[x]$.

*2) Interval Conversion:* Intervals can also be written as polynomial zonotopes. For example, let $[z] = [\underline{z}, \overline{z}] \subset \mathbb{R}^n$, then one can convert $[z]$ to a polynomial zonotope $\mathbf{z}$ using

$$\mathbf{z} = \frac{\overline{z}+\underline{z}}{2} + \sum_{i=1}^{n} \frac{\overline{z}_i - \underline{z}_i}{2} x_i, \tag{S.6}$$

where $x \in [-1,1]^n$ is the indeterminate vector.

*3) Set Addition and Multiplication:* The Minkowski Sum of two polynomial zonotopes $\mathbf{P}_1 \subset \mathbb{R}^n = \mathcal{PZ}(g_i, \alpha_i, x)$ and $\mathbf{P}_2 \subset \mathbb{R}^n = \mathcal{PZ}(h_j, \beta_j, y)$ follows from polynomial addition:

$$\mathbf{P}_1 \oplus \mathbf{P}_2 = \left\{ z \in \mathbb{R}^n \mid z = \sum_{i=0}^{n_g} g_i x^{\alpha_i} + \sum_{j=0}^{n_h} h_j y^{\beta_j} \right\}. \tag{S.7}$$

Similarly, we may write the matrix product of two polynomial zonotopes $\mathbf{P}_1$ and $\mathbf{P}_2$ when the sizes are compatible. Letting $\mathbf{P}_1 \subset \mathbb{R}^{n\times m}$ and $\mathbf{P}_2 \subset \mathbb{R}^{m\times k}$, we obtain $\mathbf{P}_1\mathbf{P}_2 \subset \mathbb{R}^{n\times k}$:

$$\mathbf{P}_1\mathbf{P}_2 = \left\{ z \in \mathbb{R}^{n\times k} \mid z = \sum_{i=0}^{n_g} g_i (\sum_{j=0}^{q} h_j y^{\beta_j}) x^{\alpha_i} \right\}. \tag{S.8}$$

When $\mathbf{P}_1 \subset \mathbb{R}^{n\times n}$ is square, exponentiation $\mathbf{P}_1^m$ may be performed by multiplying $\mathbf{P}_1$ by itself $m$ times. Furthermore, if $\mathbf{P}_1 \subset \mathbb{R}^3$ and $\mathbf{P}_2 \subset \mathbb{R}^3$, we implement a set-based cross product as matrix multiplication. We create $\mathbf{P}_1^{\times} \subset \mathbb{R}^{3\times 3}$ as

$$\mathbf{P}_1^{\times} = \left\{ A \in \mathbb{R}^{3\times 3} \mid A = \sum_{i=0}^{n_g} \begin{bmatrix} 0 & -g_{i,3} & g_{i,2} \\ g_{i,3} & 0 & -g_{i,1} \\ -g_{i,2} & g_{i,1} & 0 \end{bmatrix} x^{\alpha_i} \right\} \tag{S.9}$$

where $g_{i,j}$ refers to the $j^{\text{th}}$ element of $g_i$. Then, the set-based cross product $\mathbf{P}_1 \otimes \mathbf{P}_2 = \mathbf{P}_1^{\times}\mathbf{P}_2$ is well-defined.

*4) Slicing:* One can obtain subsets of polynomial zonotopes by plugging in values of known indeterminates. For instance, if a polynomial zonotope $\mathbf{P}$ represented a set of possible positions of a robot arm operating near an obstacle. It may be beneficial to know whether a particular choice of $\mathbf{P}$'s indeterminates yields a subset of positions that could collide with the obstacle. To this end, "slicing" a polynomial zonotope $\mathbf{P} = \mathcal{PZ}(g_i, \alpha_i, x)$ corresponds to evaluating an element of the indeterminate $x$. Given the $j^{\text{th}}$ indeterminate $x_j$ and a value $\sigma \in [-1,1]$, slicing yields a subset of $\mathbf{P}$ by plugging $\sigma$ into the specified element $x_j$:

$$\texttt{slice}(\mathbf{P}, x_j, \sigma) \subset \mathbf{P} = \left\{ z \in \mathbf{P} \mid z = \sum_{i=0}^{n_g} g_i x^{\alpha_i}, x_j = \sigma \right\}. \tag{S.10}$$

*5) Bounding a PZ:* Our algorithm requires a means to bound the elements of a polynomial zonotope. We define the `sup` and `inf` operations which return these upper and lower bounds, respectively, by taking the absolute values of generators. For $\mathbf{P} \subseteq \mathbb{R}^n$ with center $g_0$ and generators $g_i$,

$$\texttt{sup}(\mathbf{P}) = g_0 + \sum_{i=1}^{n_g} |g_i|, \tag{S.11}$$

$$\texttt{inf}(\mathbf{P}) = g_0 - \sum_{i=1}^{n_g} |g_i|. \tag{S.12}$$

Note that for any $z \in \mathbf{P}$, $\texttt{sup}(\mathbf{P}) \geq z$ and $\texttt{inf}(\mathbf{P}) \leq z$, where the inequalities are taken element-wise. These bounds may not be tight, but they are quick to compute.

*6) Set-Valued Function Evaluation:* One can overapproximate any analytic function evaluated on a polynomial zonotope using a Taylor expansion, which itself can be represented as a polynomial zonotope [1, Sec 4.1],[2, Prop. 13]. Consider an analytic function $f : \mathbb{R} \to \mathbb{R}$ and $\mathbf{P}_1 = \mathcal{PZ}(g_i, \alpha_i, x)$, with each $g_i \in \mathbb{R}$, then $f(\mathbf{P}_1) = \{y \in \mathbb{R} \mid y = f(z), z \in \mathbf{P}_1\}$. We generate $\mathbf{P}_2$ such that $f(\mathbf{P}_1) \subseteq \mathbf{P}_2$ using a Taylor expansion of degree $d \in \mathbb{N}$, where the error incurred from the finite approximation is overapproximated using a Lagrange remainder. The method follows the Taylor expansion found in the reachability algorithm in [2], which builds on previous work on conservative polynomialization found in [1]. Recall that the Taylor expansion about a point $c \in \mathbb{R}$ is

$$f(z) = \sum_{n=0}^{\infty} \frac{f^{(n)}(c)}{n!}(z-c)^n, \tag{S.13}$$

where $f^{(n)}$ is the $n^{\text{th}}$ derivative of $f$. The error incurred by a finite Taylor expansion can be bounded using the Lagrange remainder $r$ [3, p. 7.7]:

$$|f(z) - \sum_{n=0}^{d} \frac{f^{(n)}(c)}{n!}(z-c)^n| \leq r, \tag{S.14}$$

where

$$r = \max_{\delta \in [c,z]} \frac{(|f^{d+1}(\delta)|)|z-c|^{d+1}}{(d+1)!}. \tag{S.15}$$

For a polynomial zonotope, the infinite dimensional Taylor expansion is given by

$$f(\mathbf{P}_1) = \sum_{n=0}^{\infty} \frac{f^{(n)}(c)}{n!}(\mathbf{P}_1 - c)^n. \tag{S.16}$$

In practice, only a finite Taylor expansion of degree $d \in \mathbb{N}$ can be computed. Letting $c = g_0$ (i.e., the center of $\mathbf{P}_1$), and noting that $(z-c) = \sum_{i=1}^{n_g} g_i x^{\alpha_i}$ for $z \in \mathbf{P}_1$, we write

$$\mathbf{P}_2 \coloneqq \left\{ z \in \mathbb{R} \,|\, z \in \sum_{n=0}^{d} \left( \frac{f^{(n)}(g_0)}{n!} (\sum_{i=1}^{n_g} g_i x^{\alpha_i})^n \right) \oplus [r] \right\}, \quad \text{(S.17)}$$

and the Lagrange remainder $[r]$ is computed using interval arithmetic as

$$[r] = \frac{[f^{(d+1)}([\mathbf{P}_1])][(\mathbf{P}_1 - c)^{d+1}]}{(d+1)!} \quad \text{(S.18)}$$

where $[(\mathbf{P}_1 - c)^{d+1}] = [\mathtt{inf}((\mathbf{P}_1 - c)^{d+1}), \mathtt{sup}((\mathbf{P}_1 - c)^{d+1})]$ is an overapproximation of $(\mathbf{P}_1 - c)^{d+1}$. $\mathbf{P}_2$ can be expressed as a polynomial zonotope because all terms in the summation are polynomials of $x$, and the interval $[r]$ can be expressed as a polynomial zonotope as in (S.6). We denote the polynomial zonotope overapproximation of a function evaluated on a zonotope using bold symbols (i.e., $\mathbf{f}(\mathbf{P}_1)$ is the polynomial zonotope overapproximation of $f$ applied to $\mathbf{P}$).

## Supplement B
## Comparison of Robust Input Bound

ARMOUR's robust input (given in (33)) is inspired by [4], [5]. We claim that the proposed controller improves on these published controllers because it achieves the same uniform bound with a smaller robust input bound. From [4, Thm. 1, (10)], the robust input given in previous work is

$$v(q_A(t),\Delta_0,[\Delta]) = -\Big(\kappa(t)\,\|w_M(q_A(t),\Delta_0,[\Delta])\| + \phi(t)\Big)r(t), \tag{S.19}$$

where $\kappa$ and $\phi$ are positive increasing functions with $\kappa_P \geq 1$ and $\phi_P \geq 1$ as their respective minimums, and $w_M(q_A(t),\Delta_0,[\Delta])$ as in (31). With this choice of robust input, [4, Thm. 1] proves that the trajectories of $r$ are ultimately uniformly bounded by $\|r(t)\| \leq \frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}} \quad \forall t \geq t_1$, where $t_1$ is some finite time. In the context of Rem. 12, this bound holds for all time if (S.19) were used in the current framework. This uniform bound can be made identical to (35) by choosing $\kappa_P = \sqrt{\frac{\sigma_M}{2V_M}}$.

To bound the magnitude of the robust input similarly to App. E, note

$$|v(q_A(t),\Delta_0,[\Delta])|_j = \Big(\kappa(t)\,\|w_M(q_A(t),\Delta_0,[\Delta])\| + \phi(t)\Big)\,|r_j(t)|. \tag{S.20}$$

The smallest possible bound on $|r_j(t)|$ is given by $\|r(t)\| \leq \frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}}$, which yields $|r_j(t)| \leq \|r(t)\| \leq \frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}}$. Defining $w_M(*) = w_M(\mathbf{q_A}(\mathbf{T_i};\mathbf{K}),\Delta_0,[\Delta])$ as in (65), we have

$$|v(q_A(t),\Delta_0,[\Delta])|_j \leq \Big(\kappa(t)\,\|w_M(*)\| + \phi(t)\Big)\frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}} \tag{S.21}$$

for all $t \in \mathbf{T_i}$. Next, we show that a lower bound on the right hand side of this equation (i.e., a lower bound on the robust input bound) is larger than the robust input bound (66).

Using the properties of $\kappa$ and $\phi$,

$$\Big(\kappa_P\,\|w_M(*)\|\Big)\frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}} \leq \Big(\kappa(t)\,\|w_M(*)\| + \phi(t)\Big)\frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}}. \tag{S.22}$$

Cancelling terms on the left hand side gives

$$\|w_M(*)\|\sqrt{\frac{\sigma_M}{\sigma_m}} \leq \Big(\kappa(t)\,\|w_M(*)\| + \phi(t)\Big)\frac{1}{\kappa_P}\sqrt{\frac{\sigma_M}{\sigma_m}}. \tag{S.23}$$

The robust input bound for (S.19) is larger than $\|w_M(*)\|\sqrt{\frac{\sigma_M}{\sigma_m}}$.

We compare this value to ARMOUR's robust input bound in (66). Under what conditions is (66) smaller, i.e.

$$\frac{\alpha_c\varepsilon(\sigma_M-\sigma_m) + \|w_M(*)\| + w_M(*)_j}{2} \overset{?}{\leq} \|w_M(*)\|\sqrt{\frac{\sigma_M}{\sigma_m}}. \tag{S.24}$$

This relation depends on the user-specified constants $\alpha_c$ and $\varepsilon$ and $\sigma_M$ and $\sigma_m$, but generally the following are true. First, $\sigma_M \geq \sigma_m$ by definition, and usually $\sigma_M$ is much larger than $\sigma_m$. Second, $\varepsilon$ is the user-specified uniform bound and a small constant is desired to minimize tracking error. Third, the $j^{\text{th}}$ component of the worst case disturbance $w_M(*)_j$ is smaller than $\|w_M(*)\|$. Therefore, ignoring the term involving $\varepsilon$ (which is small), ARMOUR's robust input bound (66) is smaller than (S.21) by at least a factor of $\sqrt{\frac{\sigma_M}{\sigma_m}}$.

To give an idea of the differences, consider the Kinova Gen3 robot as reported in Sec. VIII-A1: $\alpha_c = 1$, $\varepsilon = 0.049089$, $\sigma_M = 18.2726$, and $\sigma_m = 8.2993$. Plugging in yields

$$\frac{\alpha_c\varepsilon(\sigma_M-\sigma_m) + \|w_M(*)\| + w_M(*)_j}{2} \leq 0.4895 + \|w_M(*)\|, \tag{S.25}$$

and

$$\|w_M(*)\|\sqrt{\frac{\sigma_M}{\sigma_m}} = 1.48380\,\|w_M(*)\|. \tag{S.26}$$

### A. Intuition Behind the Robust Input

The key intuition behind why ARMOUR's robust controller outperforms that of Giusti et al. [4] lies in how the disturbance vector $w_M(q_A(t),\Delta_0,[\Delta])$ is coupled to the modified tracking error $r(t)$. In the robust input from [4, Thm. 1, (10)], each component of the error vector is scaled by the worst-case disturbance. This means that even if uncertainty exists in only a subset of links, the robust input incorporates this worst-case disturbance at every joint, weighted by its local error. As a result, joints which actuate links that are not affected by model mismatch can still receive large robust inputs which may ultimately degrade performance. In ARMOUR's robust input (33), each component of the error vector is only scaled by the corresponding disturbance at the joint. This results in the tighter, more efficient control showing in Fig. 5.

### B. Additional Comparisons Between Robust Controllers

We further compare ARMOUR's robust controller to the controller developed in [4] by including two additional experiments. First, we compare the tracking performance of both controllers under varying model uncertainties. Assuming both controllers start with the same initial tracking error, we demonstrate that both controllers converge at a similar rate (Fig. 1). Second, we compare the robust control inputs of both controllers while tracking a reference trajectory under varying levels of model uncertainty. As the model uncertainty increases, the peak control inputs required by both controllers also increase. However, ARMOUR's control input grows 2-5x more slowly than that of [4] (Fig. 2). Together, these experiments demonstrate that ARMOUR achieves comparable convergence to [4] performance with substantially less control effort.

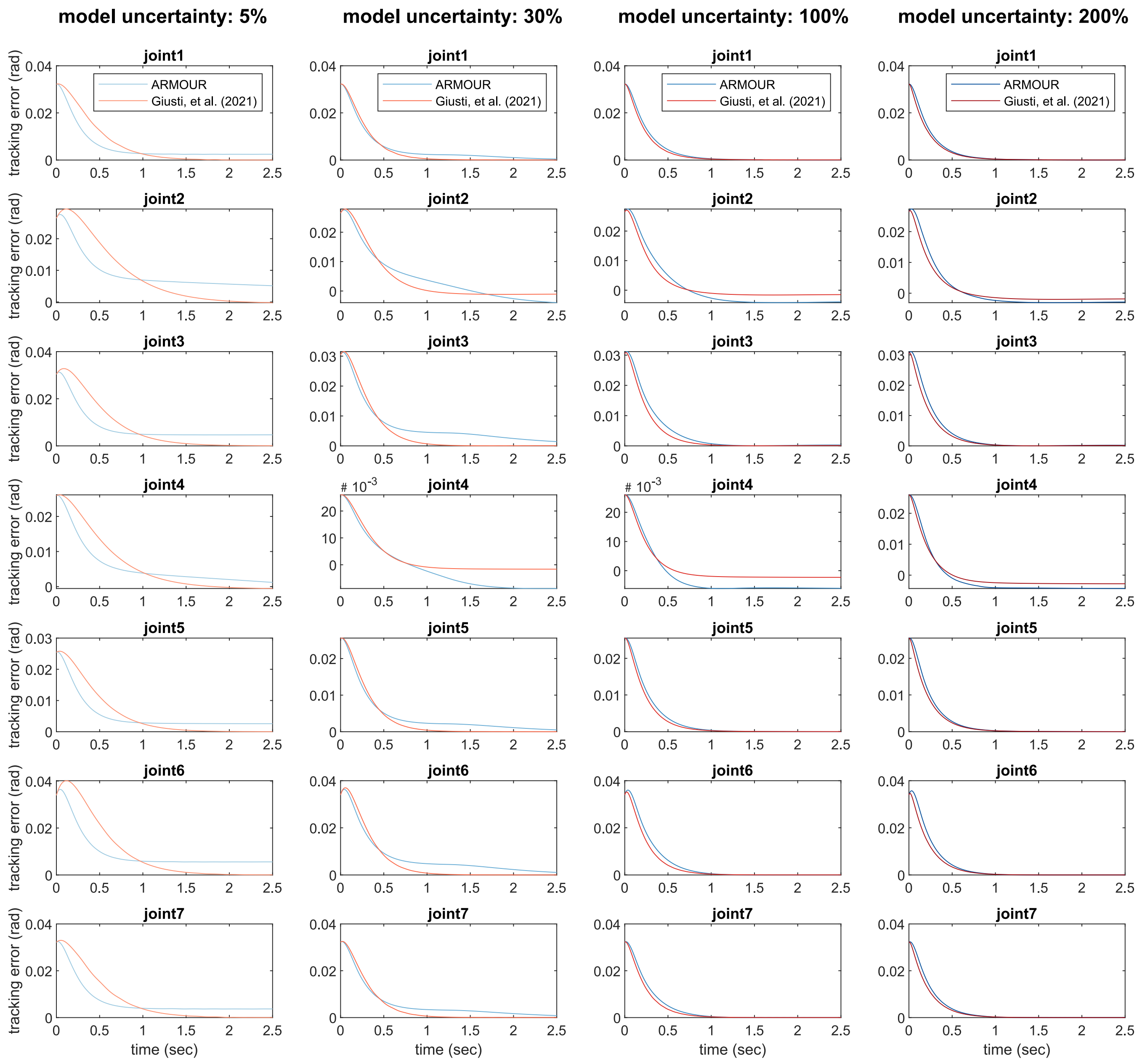

Fig. 1: This figure illustrates the tracking error of both controllers under different model uncertainties. Although both controllers start with the same initial tracking error, they converge at a similar rate, demonstrating the robustness and effectiveness of each controller.

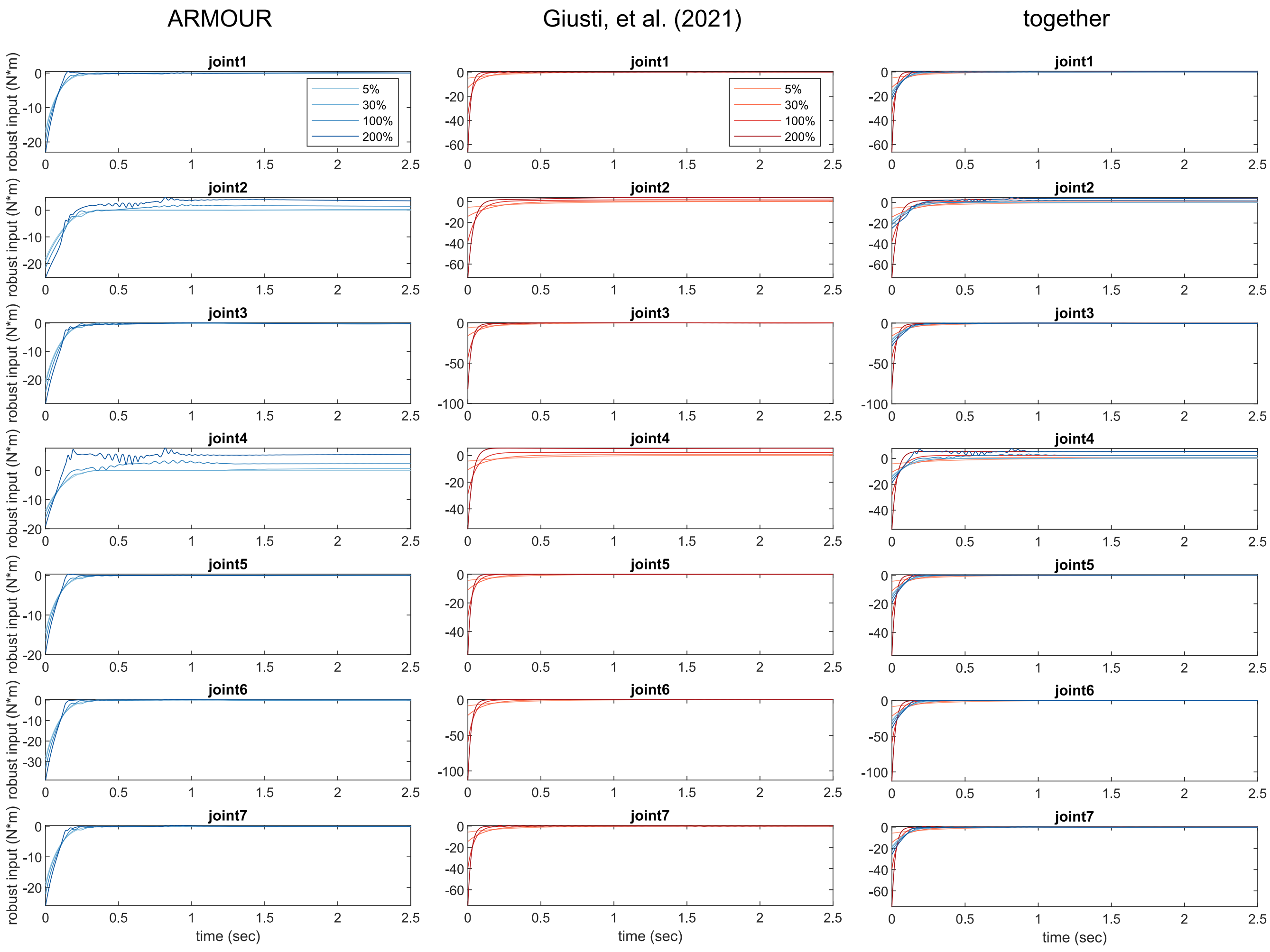

Fig. 2: This figure illustrates the robust control inputs of both controllers while tracking a reference trajectory under varying levels of model uncertainty. As the model uncertainty increases, the peak control inputs required by both controllers also increase. However, ARMOUR's control input grows 2-5x more slowly than that of [4]. This demonstrates that ARMOUR achieves comparable convergence performance with substantially less control effort.