Impact-Aware Manipulation by Dexterous Robot Control and Learning in Dynamic Semi-Structured Logistic Environments

Scenario 3 (GRAB) report

Dissemination level	Public (PU)
Work package	WP5: Integration and Scenario Validations
Deliverable number	D5.5
Version	F-1.0
Submission date	06/04/2024
Due date	31/03/2024

www.i-am-project.eu

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 871899

Authors

Authors in alphabetical order		
Name	**Organisation**	**Email**
Michael BOMBILE	EPFL	michael.bombile@epfl.ch
James HERMUS	EPFL	james.hermus@epfl.ch
Elise JEANDUPEUX	EPFL	elise.jeandupeux@epfl.ch
Harshit KHURANA	EPFL	harshit.khurana@epfl.ch
Alessandro MELONE	TUM	alessandro.melone@tum.de
Mario TROEBINGER	TUM	mario.troebinger@tum.de
Jari VAN STEEN	TU/e	j.j.v.steen@tue.nl
Ahmed ZERMANE	CNRS	ahmed.zermane@lirmm.fr

Control sheet

Version history			
Version	**Date**	**Modified by**	**Summary of changes**
0.1	06/09/2023	Jos DEN OUDEN	TOC & first contents
0.11	28/02/2024	Michael BOMBILE	General structure and first content
0.2	11/03/2024	All authors	Providing contents to sections
0.8	12/03/2024	James HERMUS	Pre-final version ready for pear-review
0.9	27/03/2024	James HERMUS	Peer-review comments addressed
1.0	06/04/2024	Jos DEN OUDEN	Revised version ready for submission, quality check

Peer reviewers		
	Reviewer name	**Date**
Reviewer 1	Maarten Jongeneel	23/03/2024
Reviewer 2	Seyed Ali BARADARAN BIRJANDI	20/03/2024
Reviewer 3	Jos DEN OUDEN	29/03/2024

Legal disclaimer

TABLE OF CONTENTS

ABBREVIATIONS

Abbreviation	Definition
EC	European Commission
PU	Public
WP	Work Package
DS	Dynamical System
QP	Quadratic Program

EXECUTIVE SUMMARY

This report presents the integration of impact-aware technologies and their application in logistics scenarios, particularly in grabbing and depalletizing tasks with dual-arm robotic systems. Impact-aware robotics leverages intentional collisions to achieve dynamic interaction and thus it has been shown to be faster and more energy efficient than traditional approaches based on quasi-static interactions with objects or environments.

The generation of desired impacts (contacts at non-zero relative speed), generally avoided in classical robotics, brings multiple challenges encompassing modeling, planning, sensing, and control to produce the desired behavior of the manipulated object, while ensuring the safety and integrity of both the robot and the object.

To tackle these challenges, the I.AM. project has developed four main groups of impact-aware technologies, referred to as *I.Model*, *I.Learn*, *I.Sense*, and *I.Control* that are combined in this work to yield a fully integrated impact-aware solution. *I.Model* provides robot-object-enviroment impact (models) law; *I.Learn* generates nominal and contingency motions with intentional impacts; *I.Sense* provides robust sensing and is capable of triggering robot reflexes in case of unexpected collisions/expected collisions with undesired outputs as fault recovery strategy; and *I.Control* provides motion execution with the ability to enforce hardware and safety constraints.

This report also highlights the benefits of the proposed approach in terms of speed and energy efficiency through extensive experiments and a systematic comparison between classical grabbing and integrated impact-aware grabbing.

1 INTRODUCTION

1.1 I.AM. project background

Europe is leading the market of torque-controlled robots. These robots can withstand physical interaction with the environment, including impacts, while providing accurate sensing and actuation capabilities. I.AM. leverages this technology and strengthens European leadership by endowing robots with the ability to exploit intentional impacts during manipulation. I.AM. focuses on impact aware manipulation in logistics, a new area of application for robotics that is expected to grow exponentially in the coming years, due to socio-economical drivers such as the booming of e-commerce and scarcity of labor. I.AM. relies on four scientific and technological research lines that will lead to breakthroughs in modeling, sensing, learning and control of fast impacts:

1. I.Model offers experimentally validated accurate impact models, embedded in a highly realistic simulator to predict post-impact robot states based on pre-impact conditions;
2. I.Learn provides advances in planning and learning for generating desired control parameters based on models of uncertainties inherent to impacts;
3. I.Sense develops an impact-aware sensing technology to robustly assess velocity, force, and robot contact state in close proximity of impact times, allowing to distinguish between expected and unexpected events;
4. I.Control generates a framework that, in conjunction with realistic models, advanced planning, and sensing components, allows for robust execution of dynamic manipulation tasks.

This integrated paradigm, I.AM., brings robots to an unprecedented level of manipulation abilities. By incorporating this new technology in existing robots, I.AM. enables shorter cycle time (10%) for applications requiring dynamic manipulation in logistics. I.AM. will speed up the take-up and deployment in this domain by validating its progress in three realistic scenarios:

- a bin-to-belt application demonstrating object tossing (TOSS scenario),
- a bin-to-bin application object fast boxing (BOX scenario)
- a case depalletizing scenario demonstrating object grabbing (GRAB scenario).

This report focuses on the development and evaluation of the case depalletizing scenario, referred to as GRAB scenario. Besides the introduction, this report is organized as follows: Section 2 describes the business value of the proposed GRAB scenario. Section 3 describes the components developed during the I.AM. project and integrated for the GRAB scenario, including the simulation environment. Section 4 presents the performance evaluation of the GRAB scenario. It starts by describing the evaluation metrics (KPIs), then the test scenarios and procedures, the two robotic setups used, and ends by presenting and discussing the experimental results conducted to validate the I.AM. GRAB scenario and highlights its benefits when compared to classical grabbing approaches. Finally, Section 5 concludes the report and provides recommendations for prospective directions.

1.2 Purpose of the deliverable

The objective of this deliverable D5.5 (GRAB report) is to describe the implementation details of the dual arm GRAB scenario demonstrator with the impact-aware technologies developed and integrated to achieve it. Moreover, the report aims to present and discuss the results of the proposed demonstrator's systematic assessment and benchmarking in comparison to the standard (almost zero-speed, impact unaware) picking and placing strategy. Additionally, the report provides prospective directions for the proposed I.AM. GRAB usage and further developments. Finally, the report also provides insight into the collaborative effort within the I.AM. consortium to achieve and validate the GRAB demonstrator. Figure 1 shows a picture of I.AM. robotics researchers from CNRS, EPFL, TU/e, and TUM during the GRAB scenario integration week organized from January 29th to February 2nd, 2024 at LASA-EPFL.

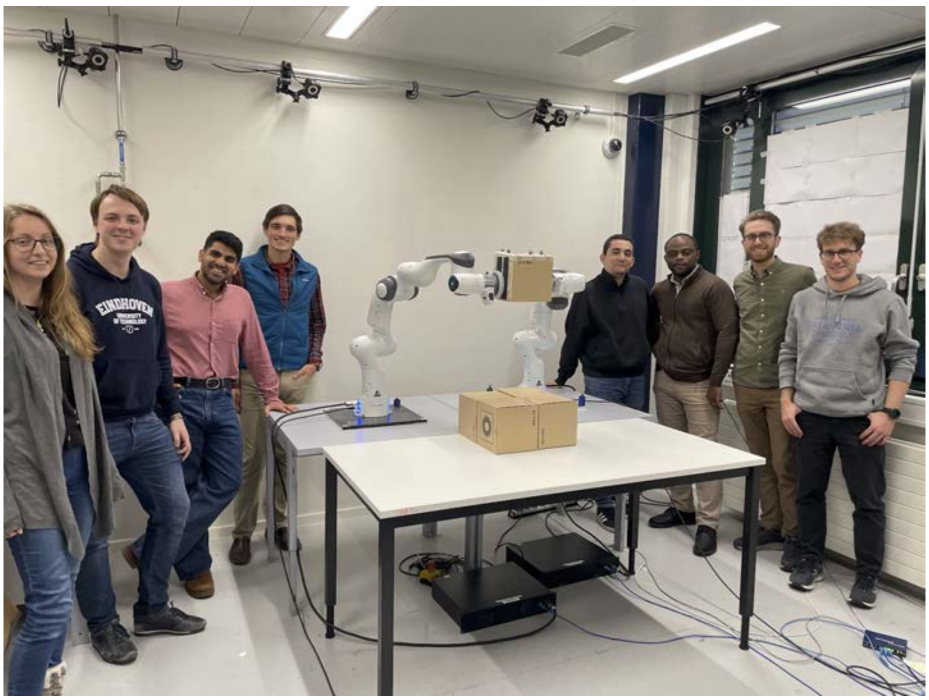


Figure 1: I.AM. robotics researchers from CNRS, EPFL, TU/e, and TUM during the GRAB scenario integration week.

1.3 Intended audience

The dissemination level of D5.5 is "public" (PU) – meant for members of the Consortium (including Commission Services) and the general public. This document is designed to serve as an internal communication for the entire I.AM. consortium by providing information on the integration of developed impact-ware technologies for the case depalletizing scenario demonstrating object grabbing.

2 Business value

2.1 GRAB value

This report focuses on one industrial use case in logistics that would benefit from the exploitation of impacts. It considers dynamic grabbing in the context of depalletizing tasks using robotic dual-arm systems. Depalletizing is a process where objects are picked from pallets and placed or tossed on conveyor belts, trays, or in tote containers.

In logistics, the booming of e-commerce and its related challenges have increased the need to speed up the pace of pick-and-place operations. Although there exist fully automated depalletizing solutions that often use specialized equipment or single robotic arms with tools adapted to the types of products to be depalletized, the current dual-arm solutions to depalletizing rely mainly on human operators. The specialized automated systems are suitable for high volume with little variety in product types [1]. They have, however, large footprints when compared to manual depalletizing solutions. These are best suited for low volume with a wide variety of products since humans can effortlessly adapt to such high-mixed environments. There also exist robotic solutions with smaller footprints and higher versatility, for instance using cobots as in [2]. However, these systems have low throughput compared to humans mainly because of their quasi-static interaction approaches. As for manual solutions, given the physical demands of these tasks associated with the scarcity of labor, the current workforce, although having better dexterity and flexibility, cannot keep up with the growing industry needs.

Thus, the goal of this report is to develop an integrated framework of technologies developed during the European Union-funded impact-aware manipulation (I.AM.) project to produce an autonomous case depalletizing solution based on a dual-arm robotic system, capable of grabbing and releasing objects with non-zero relative velocities. The desired manipulation task is motivated by the need to perform faster (10%) pick-and-place or pick-and-toss operations in a depalletizing context using a versatile system with a smaller footprint than the current automated solutions.

2.2 Potential Work environment

Our goal is not to offer a solution fully integrated into the current industry. Instead, we aim to showcase how integrating the technology developed within the consortium can leverage impact and accelerate the depalletizing process. This demonstration will take place in a laboratory environment using robotic dual-arm solutions closely resembling industrial setups.

The work environment that we will mimic for the robotic dual-arm solution. Vanderlande Industries has identified the environment where the robotic dual-arm solution could potentially demonstrate its value. In figure 2 and 3 the work environment is shown.

One can see that items arrive one layer at a time (so items are not stacked on top of each other), also the items are closely stacked together with little to no space in between. Examples of what the item set consists of are carton boxes, plastic-wrapped bottles, and beer cases, which are difficult to pick using a vacuum gripper. Every layer of items comes with a layer of carton on top, removing this layer of carton will be out of scope for the I.AM. project. After picking the items, they must be placed in a tray, the bottom of the tray will move up, to make placement easier. Multiple items will be placed in the tray, the number of items that need to be placed in the tray depends on the type of item.

Figure 2: A depalletizing station operated by humans, items are provided one layer at a time and are closely placed together with little to no space in between. Items are picked by the operator and placed on a belt.

(a) (b)

Figure 3: At a depalletizing station operators might pick multiple items, and either lift them into the tray or slide them over the surface of the station into the tray

3 Developed Components for GRAB Scenario

The preceding discussion on the industry environment has served as a catalyst for our Grab integration scenario. In this study, we aim to showcase and evaluate the technology developed within our consortium, particularly in the context of the dual-arm pick-and-toss task. Unlike the prevailing industry practice of cautiously executing quasi-static contacts for both picking and placing objects, our approach emphasizes the rapid initiation and termination of contact. This capability enables a significant enhancement in task efficiency.

To illustrate, we will demonstrate a swift dual-arm pick and toss maneuver. Although dual-arm systems are not yet prevalent in industry, we will benchmark our approach against the industry-standard quasi-static pick-and-place method. This comparison will be detailed further in Section 4.

We describe the components developed and integrated to achieve impact-aware dual-arm grabbing. We start by presenting the control architecture showing the components and their interconnections. These components discussed later on are namely the GRAB: i) impact planner, ii) impact-aware motion and force generator, iii) impact-aware controller, iv) contact state estimator, and v) simulation environment.

In the GRAB scenario presented in Section 4 the motion generation and force generation from Section 3.2.1 (I.Learn), the reference adaptation for impacts from Section 3.2.2 (I.Model), the impact-aware controller from Section 3.3 (I.Control), and contact state sensing from Section 3.4.1 (I.Sense) were integrated in hardware at EPFL.

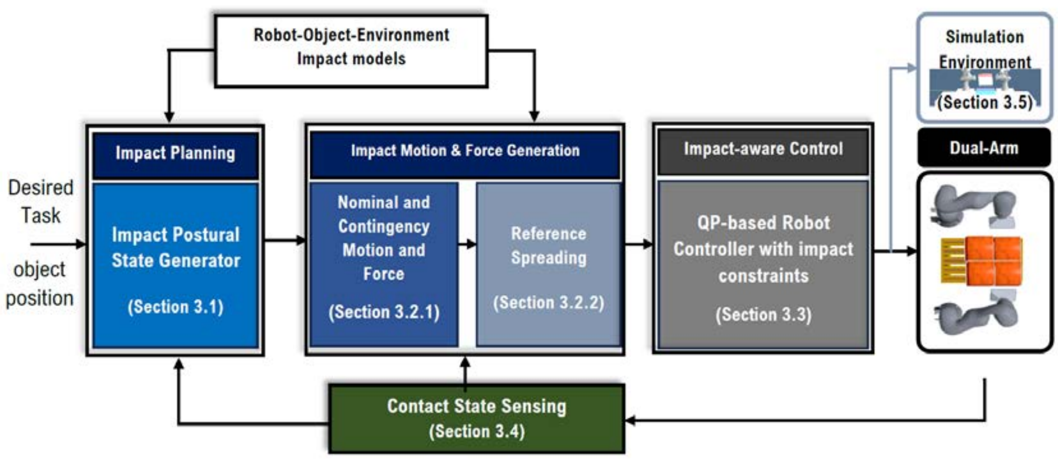


Figure 4: Block diagram of impact-aware GRAB components

3.1 GRAB Impact Planning

This subsection describes the developed impact planning strategies whose goal is to determine the dual-arm impact posture, direction, and speed that will lead to the desired post-impact state while satisfying the robot-object hardware constraints.

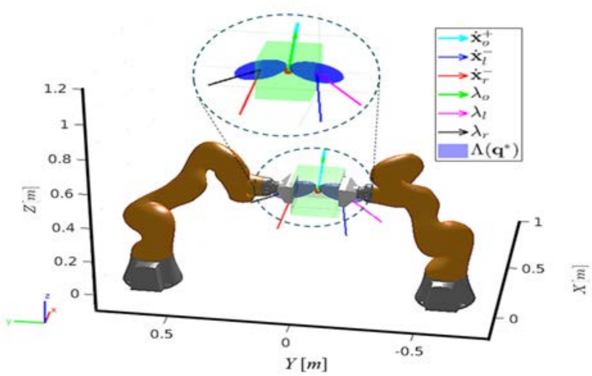
3.1.1 Dual-arm Impact State Generator

To determine the desired impact state of an isolated object, we devise an impact planner that computes for given desired grabbing points and the desired post-impact state of the object, the corresponding dual-arm impact posture, and associated minimal pre-impact velocities (directions and magnitudes) subjected to the robots' joint velocity and torque limits. To achieve its goal, the proposed planner follows an optimization approach and assumes an inelastic impact between the two robots and the object taken in a sandwich. It exploits the impact map models of each robot in conjunction with the multi-body collision model, and the object's effective force distribution between the two robots under unilateral frictional contact constraints.

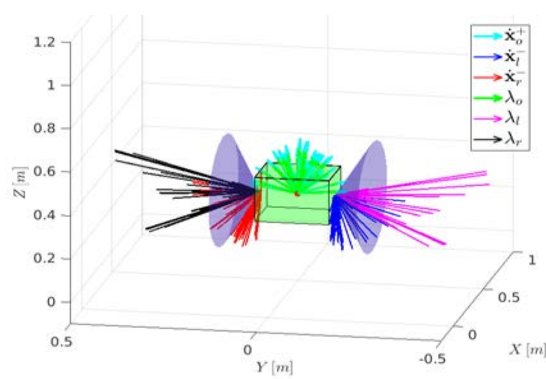
The overall method is iterative and can be summarized in the following steps:

- **Step 1:** compute the desired posture corresponding to the given object's grabbing points.
- **Step 2:** given the posture, compute the constraint-consistent minimal pre-impact velocities and impact forces corresponding to the desired post-impact velocity of the object and dual arm.
- **Step 3:** from the pre-impact velocities, determine the impact directions, go to **Step 1**, and update the dual-arm posture by optimizing the task space inertia along the obtained impact directions.

The above steps are repeated until convergence (no change in impact directions).
An example of an impact posture with associated task space impact velocities and forces of a sample of 50 desired post-impact states are illustrated in Figure 5. The joint space and task space impact state variables corresponding to the selected impact posture are shown in Figure 6.



(a)

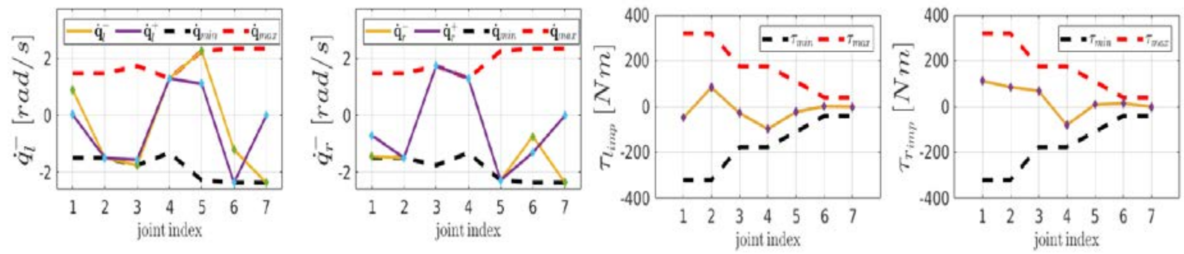


(b)

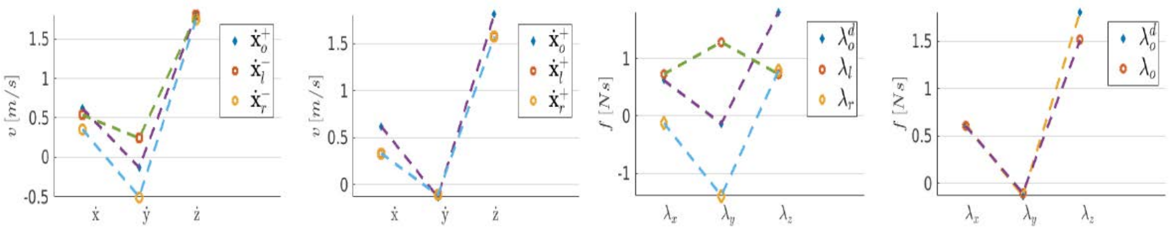
Figure 5: Example of a feasible impact posture of the dual-arm system with corresponding end-effector inertia ellipsoid. a) impact velocities and forces direction in 3D. b) Illustration of 50 computed optimal feasible impact states (velocities and forces) with task space velocities $\dot{\mathbf{x}}_0^+$, $\dot{\mathbf{x}}_l^-$, $\dot{\mathbf{x}}_r^-$ in $[m/s]$ and the task space forces $\boldsymbol{\lambda}_0$, $\boldsymbol{\lambda}_l$, $\boldsymbol{\lambda}_r$ in $[Ns]$

3.1.2 Dynamical systems for hitting

EPFL has worked on a combination of linear DS that generate motion to hit / quickly push an object and hence place it outside the workspace of the robot. The impact generated upon contact with the object



(a)



(b)

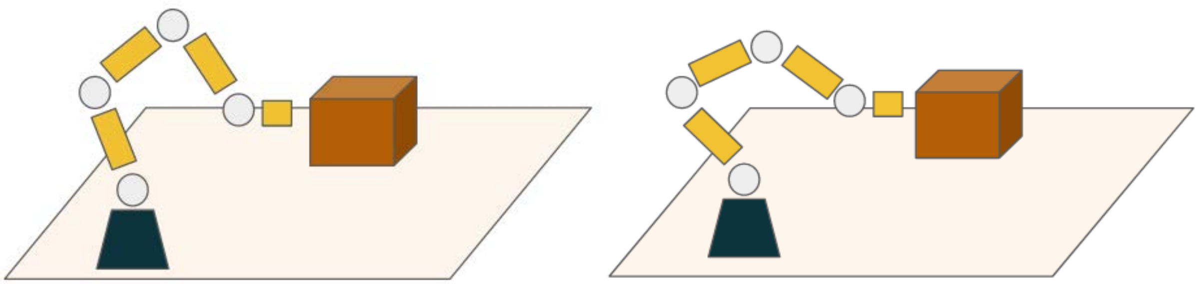
Figure 6: Joint space and task space impact state variables corresponding to the impact posture of Figure 5 a) pre- and post-impact joint velocity and torque with their hardware limits. b) task space pre-impact velocities and post-impact velocities and forces

is at high velocity, thus quasi-static frameworks of robot-object contact cannot be used. The motion generation for the robot depends on the hitting parameters - namely the hitting speed of the robot and the hitting direction. These are determined with the model of the motion of the object on a planar surface. Since the motion of the object is a function of the robot's motion, the point of hit, the friction between the object and the surface, impact with the robot etc., simple analytical methods are not able to capture this complexity. Currently, we chose data driven approach for determining the motion of the object. This made it possible to determine the hitting parameters for the desired final position of the object. The data is first modelled using a Gaussian mixture model and then through regression using the fore-mentioned model, the hitting parameters for the desired location of the object are predicted. These are then used to generate the motion of the robot and the results of the framework are tested on KUKA iiwa lbr 7 arm. The motion is controlled using a passive controller that allows for human interaction along the path generated by the DS. Further work on determining the evolution equation of the robot's configuration which is approaching the object to make contact at non-zero velocity has been done. Since there are multiple solutions to the configuration achieved by the robot, there exists a problem formulation to not just identify at what configuration should the robot come in contact but also how the robot achieves the configuration while maintaining the path of its end effector. This fact is shown in Figure 8.

To keep track of the inertia of the robot through its motion and to optimize the configuration of the robot, we have used the task space inertia matrix, the DS for motion generation uses a virtual end effector position along the hitting direction. This helps identify the directional inertia of the robot which is used in the control strategy proposed to control the inertia of the robot. In the latest research in hitting/striking/poking an object, EPFL has proposed a new entity to control during the motion of the robot, called **hitting flux** [3]. The hitting flux takes into account the desired post-impact properties of

(a) Approaching phase (b) Hitting phase (c) Final position of the object

Figure 7: Snapshots depicting three stages of the box hitting experiment. In (a) the robot is approaching the object. In (b) the robot hits the object. We can see here that the impacting with a flat end-effector leads to further uncertainty due to differences in orientation of the box and the end-effector. In (c) we see the final position of the object being outside the physical workspace of the robot.



(a) Reaching object with configuration 1 (b) Reaching object with configuration 2

Figure 8: There are possibly infinite ways in the joint configuration to reach the object. The configuration changes the inertia of the robot and hence raises the issue of optimizing the reach configuration.

the object.

From the basic collision equations - the principle of conservation of momentum and the coefficient of restitution, we can derive the following equations:

$$\Lambda\dot{\chi}_e^- + M_o\dot{\chi}_o^- = \Lambda\dot{\chi}_e^+ + M_o\dot{\chi}_o^+ \tag{1}$$

$$E[\dot{\chi}_e^- - \dot{\chi}_o^-] = \dot{\chi}_o^+ - \dot{\chi}_e^+ \tag{2}$$

where, Λ is the inertia of the robot, M_o is the mass of the object, $(\dot{\chi}_o^-, \dot{\chi}_e^-), (\dot{\chi}_o^+, \dot{\chi}_e^+)$ are the pre and post-impact velocities of the object and the end effector of the robot respectively, and E is the coefficient of restitution. The following assumptions simplify the above equations:

- The pre-impact velocity of the object, $\dot{\chi}_o^- = 0$. Hence, the pre- impact robot velocity is

  $$\dot{\chi}_e^- = (\Lambda[I + E])^{-1}([\Lambda + M_o]\dot{\chi}_o^+) \tag{3}$$

  where I is the identity matrix.

- The robot-object contact is instantaneous with no contact after the collision. The distance between the robot end effector and the object is greater than zero.

- The configuration and hence $\Lambda(q)$ of the robot remains constant during the collision.

Let $(I + E) = C$, and we can write the post impact object velocity as

$$(I + \Lambda^{-1} M_o)^{-1} C \dot{\chi}_e^- = \dot{\chi}_o^+ \tag{4}$$

Thus, to achieve similar post-impact velocity on the object, the quantity to be controlled is given by equation 4 and this depends on inertia of the robot, the end effector velocity and the mass of the object, and we label this quantity as hitting flux, Φ.

$$(I + \Lambda^{-1} M_o)^{-1} C \dot{\chi}_e^- = \Phi \tag{5}$$

The control strategy for controlling the inertia of the robot is elaborated in the following subsection. With the control system leading to directional inertia values close enough to the desired values, here we formulate a strategy to achieve the desired directional flux, ϕ^*. We have:

$$\phi = (1 + m/\lambda_h(q))^{-1} \dot{\chi} \tag{6}$$

To achieve the desired directional flux, ϕ^*, the desired velocity, $\dot{\chi}^*$ in the DS $f(\chi)$ depends on ϕ^*. [1]

$$\dot{\chi}^* = \phi^* [1 + m/\lambda(q)] \tag{7}$$

The DS can be seen in the experiment video linked here. The code for the DS with the robot simulation is publicly available here. This work was published in IEEE T-RO [4].

The existing QP-control framework performs one-step-ahead prediction only and cannot plan trajectories in advance. On the other side, learned DSs encapsulate demonstrations and therefore replace time-consuming trajectory re-planning at run-time. In a joint CNRS-EPFL effort, a robot-independent interface is developed for integrating task-space DS within the QP controller. Both first-order and second-order DSs are supported. The task-space DS is integrated as a quadratic objective that is to be minimized. The resulting framework combines the advantages of both worlds: the robot follows pre-planned task-space trajectories and is able to react to disturbances without requiring additional computations. The quadratic program (QP) replaces the classical inverse kinematics solver used at EPFL, providing advanced (self-)collision avoidance while respecting the robot's hardware limits. The integration does not require any meta-parameters when using a second-order DS as the QP employs accelerations as decision variables. Due to the chosen generic structure, the DS integration applies to all three existing robot control modes: joint position control, joint velocity control, and joint torque control. New DSs to be developed by EPFL or updated DS with additional data can now easily be used within I.Control. The novel DS-based motion control has been demonstrated in RVIZ, Choreonoid, and AGX Dynamics simulation as well as with the real panda robot.

3.1.3 Controller for robot hitting

The hitting DS is a function that provides the desired end effector velocity of the robot given its position and the directional inertia (described in Task 2.3.1a). This direction is the direction of hitting. If we are aware of a desired directional inertia value or a property of the desired directional inertia such as it should be locally maximum, then we can add these desired tasks to the Null space of the controller. This

[1] All the directional quantities are represented with the lowercase Greek letters.

allows the robot to move along the path and allows the inertia to possess the desired property, if feasible. Let, $\lambda_h(q) = \hat{h}^T \Lambda_t(q) \hat{h}$ be the directional inertia, where $\hat{h}$ is the unit vector in the hitting direction, and $\Lambda_t(q)$ is the translational task space inertia of the robot at its end effector. As the robot end effector moves along $f(\chi)$ (the DS), the following controller allows for increasing the inertia perceived at the end effector along the direction of the vector field $f(\chi)$

$$\dot{q} = J^{\dagger}(q) f(\chi) + N[\beta_1(q_m - q) + \beta_2 \nabla_q \lambda_h(q)], \quad \beta_1, \beta_2 \in \mathbb{R}_+$$

where, $N = I - J(q)^{\dagger} J(q)$ is the dynamically consistent Null space for joint velocities, and β_1, β_2 are hyperparameters, which control the relative weights of achieving high manipulability configuration and moving in the direction of increasing directional inertia.

Instead of maximizing the directional inertia, if we want to achieve a desired directional inertia λ^* while tracking the DS, with priority to the latter. To control for the directional inertia in the null space, $\nabla_q \lambda_h(q)$ is multiplied with $(\lambda_h(q) - \lambda^*)$. This changes the joint velocities towards the configuration with the desired directional inertia if the redundancy allows for it.

$$\dot{q} = J^{\dagger}(q) f(\chi) + N[\beta_1(q_m - q) + \beta_2(-\nabla_q \lambda_h(q)(\lambda_h(q) - \lambda^*))], \quad \beta_1, \beta_2 \in \mathbb{R}_+$$

The effect of the above controllers can be seen in the figure 9. The increasing directional inertia controller tries to align the entire robot in the direction of hitting, which the specific directional inertia controller leads to a joint configuration that controls how much of the robot gets aligned in the hitting direction.

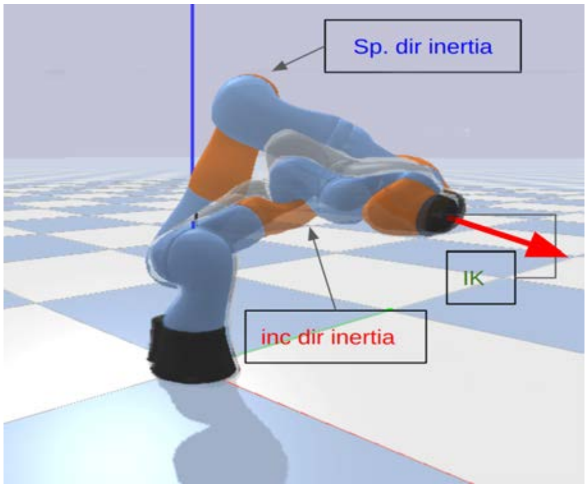
One step further from the directional inertia, we also move in the direction of achieving the desired translational inertia matrix (Λ^*). Stein distance is used as a distance metric between the current and desired inertia matrix, represented as $g(q)$. We minimize the distance between the current inertia matrix and the desired inertia matrix, while following a DS under joint position and velocity constraints.

$$\begin{aligned} \dot{q} = &\underset{\dot{q}}{\operatorname{argmin}} \quad \frac{1}{2} \left\| f(\chi) - J\dot{q} \right\|_2^2 + k_1 (\nabla_q g(q)) \dot{q} + k_2 \left\| \dot{q} \right\|_2^2 \\ \text{s.t.} \quad & \dot{q}_{min} \leq \dot{q} \leq \dot{q}_{max}, \quad q_{min} \leq q \leq q_{max} \\ & k_1, k_2 \in \mathbb{R}_+ \end{aligned} \tag{8}$$

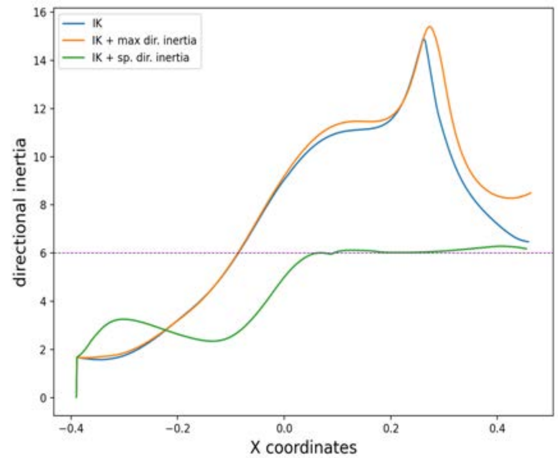
where, k_1 and k_2 are the weights for the penalties on derivative of the stein distance and the joint velocity norm. Figure 10 shows that following the inertia based controller leads to different configurations, the inertia of which is closer to the desired inertia matrix. The code for the controllers with the robot simulation is publicly available here.

3.1.4 Hit to tilt

Tilting objects is a form of non-prehensile robotic manipulation [5]. It can be handled using slower dynamics; for example, [6] used a suction cup end-effector to tilt an object by developing a framework that withstands deformation modeling errors, which was also studied in detail within the I.AM. project in the BOX scenario [7] with enhanced modeling of the suction cup deformation [8]. In [9], a robust pivoting formalism accounts for object inertia parameters. It is developed based on contact implicit bilevel optimization and enhanced by a tactile closed-loop control to deal with initializing issues and

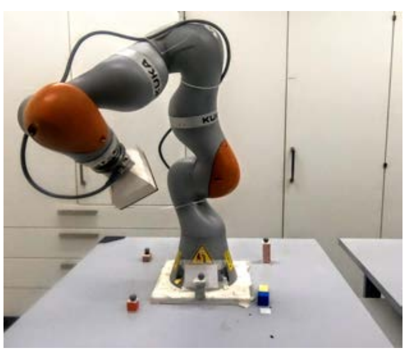


(a) Directional inertia controllers' effect

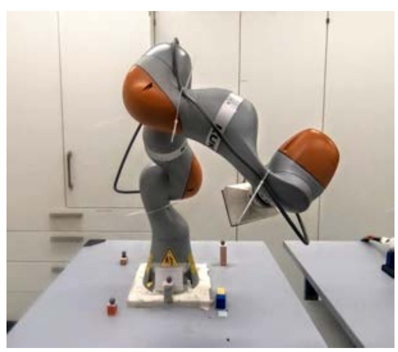


(b) Directional inertia of the robot during the trajectory

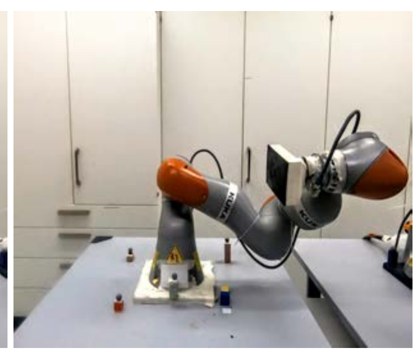
Figure 9: Figure (a): The three different control systems lead to different joint configurations at the desired final position. The inverse kinematics controller leads to the joint configuration closest to the initial configuration. With the increasing directional inertia controller, the robot's configuration is aligned with the desired direction. Since the specific directional inertia is lower than the maximum achievable in the desired direction and the directional inertia achieved by the inverse kinematics controller, the configuration achieved with the specific inertia controller is least aligned with the hitting direction; Figure (b): Quantitative comparison of the controllers - The three curves show the different directional inertia of the robot while following a trajectory with three different controllers. The green curve shows the controller tries to maintain the directional inertia along the trajectory equal to 6 units while following the trajectory.



(a) Initial position of the robot



(b) Motion with inverse kinematics Controller



(c) Motion with Inertia QP Controller

Figure 10: Figure (a): This is the starting configuration of the robot; Figure (b): Final configuration of the robot under the inverse kimematics Controller. This configuration is moving away from the desired inertia matrix; Figure (c): Final configuration of the robot under the Inertia QP Controller. This configuration is closer to the desired inertia configuration while achieving the final position

enable failure recovery. In this section, we were interested in achieving it using impacts and by applying an impulse on an object's surface to induce an initial velocity that brings it to a desired state. We use an explicit model-based plugin that integrates our planning strategy by accounting for all forces acting on the box; see Fig. 11.

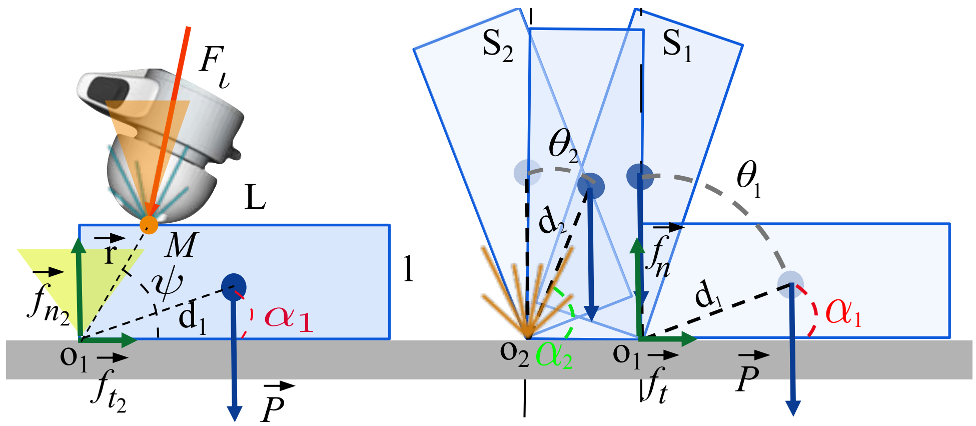


Figure 11: On the left side, the force applied on the box exactly before starting to tilt. The box and applied forces during tilting phases S_1 and S_2 are on the right one.

At impact time T, utilizing Newton's Laws circumvents the complexity linked to state discontinuities [10]. Thus, there is a need to use second-order Measure Differential Equation (MDE) (assuming no sliding):

$$\mathcal{I}_{\mathrm{yy}/o_1}\dot{\omega} = \vec{\tau}_\iota \delta_T \vec{j} + \vec{\tau}_P \vec{j} \tag{9}$$

$$\begin{aligned} \text{and} \quad \vec{\tau}_\iota &= \boldsymbol{v}OM \times \vec{F}_\iota = [0, -z_M f_{\iota_x} + x_M f_{\iota_z}, 0]^T \\ \vec{\tau}_P &= \boldsymbol{v}OG \times \vec{P} = [0, mgx_G, 0]^T = [0, mgd_1\cos(\alpha_1), 0] \\ \boldsymbol{v}OM &= \left[l\cot(\psi), 0, l\right]^T, \boldsymbol{v}OG = \left[d_1\cos(\alpha_1), 0, d_1\sin(\alpha_1)\right] \end{aligned}$$

are the coordinates of the impact point M and the object's CoM; $\vec{F}_\iota$ is the impact force generated by the robot end-effector; $\vec{P}$ is the gravity force acting on the box; and δ_T is the Dirac measure at impact. $[\mathcal{I}, \dot{\omega}]$ are the box inertia matrix and the angular acceleration. We enforce non-sliding impact with

$$|\vec{f}_t| \leq \mu_0 f_n, |\vec{\tau}_\iota| \geq D_y f_n, \text{and} f_{\iota_z} > f_{\iota_x} > 0 \tag{10}$$

where μ_0 and D_y are friction coefficients. Post-impact dynamics can be expressed as

$$\mathcal{I}_{\mathrm{yy}/o_1}\dot{\omega} = \vec{\tau}_P(\theta)\vec{j} = mgd_1\cos(\theta)\vec{j} \tag{11}$$

with $\omega_0 = \omega_0^+ \neq 0$ resulting from the impact, and $\theta^+ = 0$.

the post-impact velocity at M is written as:

$$\vec{v}_M^+ = \vec{\omega}^+ \times \boldsymbol{v}OM = \omega^+[l\cot(\psi), 0, l]^T \tag{12}$$

we get ω^+ by solving the energy-based equation eq. 13:

$$\frac{1}{2}\mathcal{I}_{\mathrm{yy}/o_1}\omega^{+^2} = mgd_1(1 - \sin(\alpha_1)) \tag{13}$$

$$\omega^+ = \sqrt{\frac{2mgd_1}{\mathcal{I}_{\mathrm{yy}/o_1}}(1 - \sin(\alpha_1))} \tag{14}$$

we can then estimate the impact point and velocity from eq. 12 to plug it into the impact planner. However, there are situations where, even in perfect physics, reaching a settled pose is impossible. The right amount of velocity to bring the object to reach its intended pose induces a rotational trajectory from the initial resting pose to the singular point at near-to-nil angular velocity (S_1 in Fig. 11). The CoM position to the rotation axis can make this task difficult to achieve. By analyzing energy at singularity positions S_1 and S_2 (see Fig. 11), we set two main conditions: (i) $\theta_2 + \alpha_2 \leq \frac{\pi}{2}$, and
(ii) $E_2 = mgd_2\cos(\theta_2 + \alpha_2) \leq \epsilon E_1 = \epsilon mgd_1\cos(\alpha_1)$, where ϵ is the coefficient of restitution.

Assuming that, at S_1, $\omega_{S_1} \geq 0$; at S_2, $\omega_{S_2} = 0$; and $\epsilon = 1$,

$$mgd_1 = mgd_2\cos(\theta_2 + \alpha_2) \tag{15}$$

since $\cos(\theta_2 + \alpha_2) \in [0, 1]$ we have

$$d_1 \leq d_2 \tag{16}$$

which means $\frac{\pi}{2} - \alpha_2 \leq \alpha_1$. We tested tilting different boxes utilizing a 3D-printed rigid end-effector.

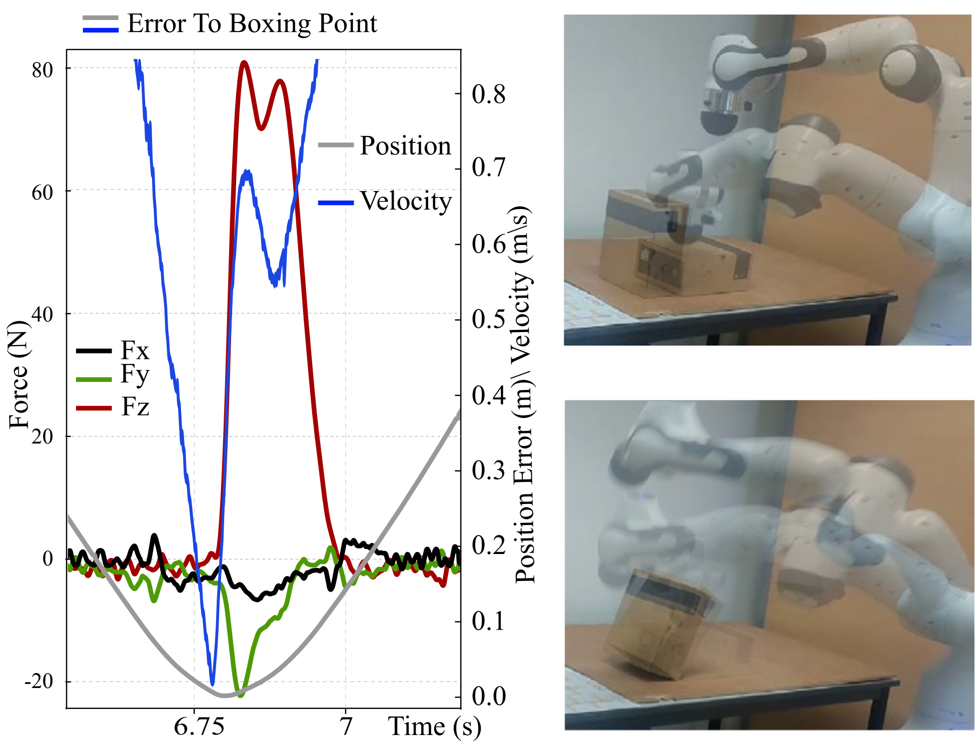


Figure 12: Boxing sequence (right) and measured impact forces (consequent to desired impacting velocity) with position/velocity error to impacting point M (left).

Figure 12 shows the execution of the planned trajectory. The desired impact point M is reached with the corresponding velocity to induce the intended planned rotational motion. Figure 13 illustrates the velocity tracking performance in the boxing experiment (namely in the presence of velocity jumps produced by the impact), and how important it is to take the robot back to a safe configuration to proceed to with the next task objective.

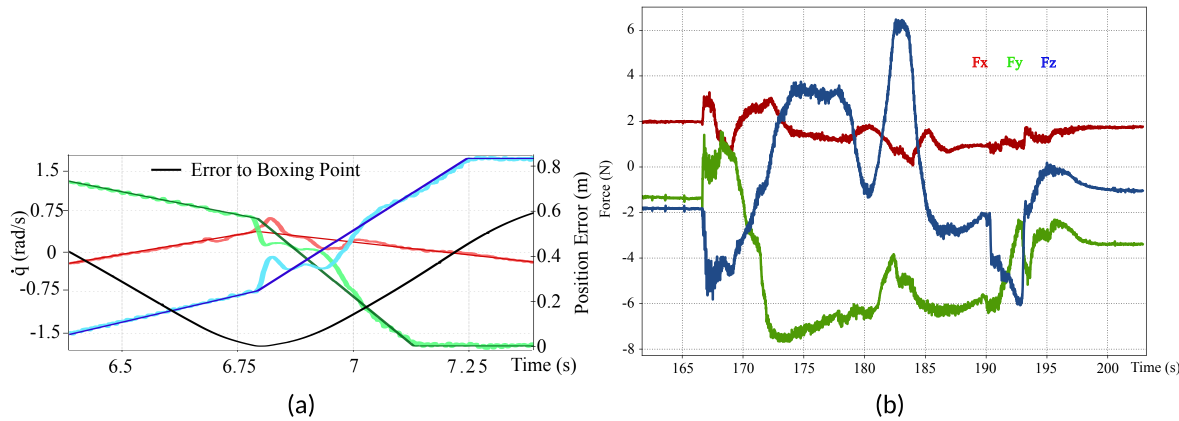


Figure 13: Figure (a):Desired joint velocity for three joints, $i = \{2, 3, 4\}$, dark colors (red, blue, green) and the real measured corresponding velocities by the encoders with light colors. Figure(b): Sustained contact force measurements while tilting box.

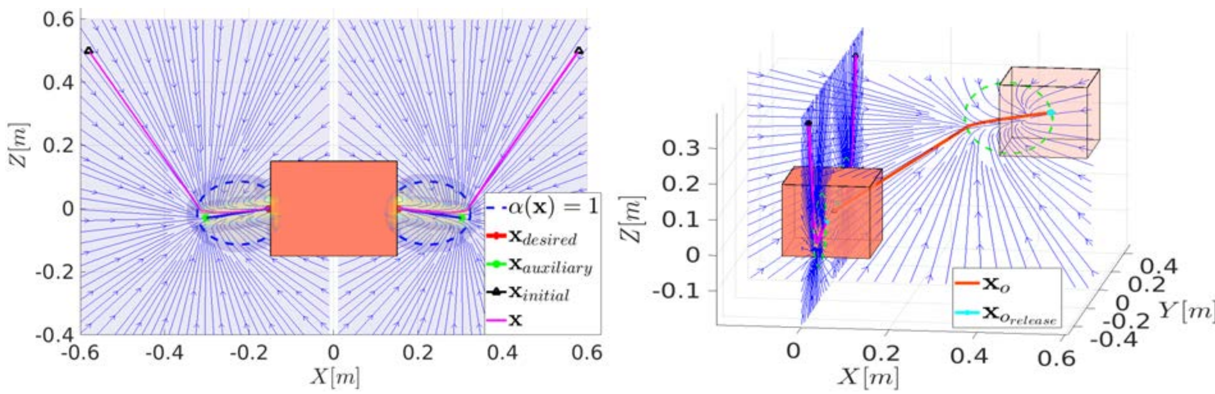
In case of boxes that cannot reach the desired pose in a stand-alone motion (i.e., would bypass S_2 in Fig. 11), it is possible to plan another robot arm to meet the box a little after the singular configuration S_1 in Fig. 11 and still achieve boxing.

Indeed, the tilting studied in this case is an extreme dynamic motion case; in most cases, it is acceptable to gently tilt boxes to grab them with slower dynamics by applying a continuously sustained contact. This can be done as formulated in [11] using tactile feedback, where contact is sustained during the whole motion. In terms of performance, rotating a box in [11] took ~ 160 s. Our implementation of contact-sustained boxing using force sensing took 45 s see Fig. 13, whereas impact-boxing is made in around 1 s!

3.2 GRAB Motion and Force Generation

3.2.1 Pre-impact and Post-impact motion

To achieve pre- and post-impact motion with a dual-arm system in the GRAB scenario, we propose a unified motion generator that enables a bimanual robotic arm system to grab and lift/place/toss an object in one swipe [12]. This continuous control of reach, grab, and toss motion is achieved by combining a sequence of time-invariant DSs in a single control framework. From a control perspective, a desired impact or tossing state represents an intermediate or transitory state defined in terms of desired position and velocity that must be satisfied simultaneously at the impact or release instant. The requirements of such a task are fulfilled by adopting, more precisely, a modulated DS approach where state-dependent modulation functions locally shape the motion of the robot such that it passes through the desired impact states. The main idea is to generate motion towards an attractor located near the desired release position, and when in its vicinity (within the modulation region), reshape the robot's motion - prior to contact or release of the object - such that the motion aligns first with the desired velocity direction while moving towards the desired contact or release position. In addition to controlling for impact, we also control the coordination between the robotic arms to ensure the success of the dual-arm grabbing task. A poorly coordinated system, where one arm reaches the object before the other, would lead not only to uncontrolled impact but also failure of the post-grabbing task. Thus, we use the cooperative control framework for the motion and use a QP-based scheme to generate constraint-consistent contact forces to stabilize the grabbing task. Illustrations of the proposed DS motion flow when grabbing with impact and tossing are shown in Figure 14.



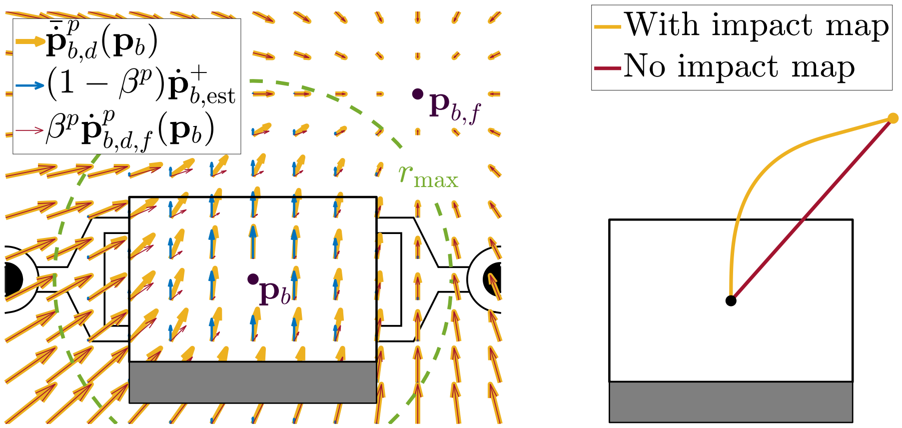
(a) (b)

Figure 14: Illustration of motion flow generated by the DS outside and within the modulated region. (a) motion of each robot is shaped within the modulation region (thick dotted blue line) such that it passes through the desired transitory state (here an impact state) with the desired position represented by the red dot and the direction of the desired velocity represented by the blue arrow. (b) object's motion flow generated by the DS once the object is grabbed and carried by the dual-arm system.

3.2.2 Reference adaptation for impacts

Performing motions with impacts implies a velocity jump for both robots. If the pre-and post-impact velocity references do not capture this velocity jump correctly, switching from the pre-impact reference to the post-impact reference could trigger a sudden large velocity tracking error. This can in turn result in jumps in the control inputs that can trigger vibrations, damage or destabilize the system, and increase energy consumption. Therefore, the pre- and post-impact velocity references are adapted in order to match the predicted post-impact velocity jump.

This adaptation is in line with the reference spreading approach, originally introduced in [13] and adapted throughout this project in [14, 15] to fit with time-invariant velocity references such as those obtained from the approach described in Section 3.2.1. In this approach, numerical simulations with AGX Dynamics using the GLUE framework developed throughout the project are performed, initializing the system in a range of possible impact locations. The resulting post-impact velocities are then saved and used to locally modify the post-impact velocity reference obtained from the DS-based approach presented in Section 3.2.1. An example of this local modification for a 2D use case is shown in Figure 15. Whenever the manipulated object is within a given distance from its initial position, a convex combination of the predicted post-impact velocity and the post-impact DS is used as post-impact reference. This results in a more natural post-impact motion, without input spikes at the time of transitioning to the post-impact mode.



(a) Post-impact velocity reference field.

(b) Nominal box path.

Figure 15: 2D example of post-impact reference adaptation for impact tasks (figure taken from [15]). The post-impact reference matches a pre-impact reference where both robot grasp the box with an upward velocity.

3.3 GRAB Impact-Aware Controller

To enforce the reference motion described in Section 3.2, the mc_rtc[2] control framework, developed by CNRS, is used. This framework uses a Quadratic Programming (QP) optimization problem to generate desired joint torques for both robots using a constrained optimization problem. These torques are then sent as input for the robots' low-level torque control. The QP constraints ensure the robots do not violate joint limits and prevent (self-)collisions, while the cost function is created using tasks that enforce the reference motion and force described in Section 3.2. Additionally, a posture task is added to resolve the kinematic redundancy of the 7DOF robots.

Three different control modes are defined: 1. an ante-impact mode, 2. an interim mode, and 3. a post-impact mode. The ante-impact mode is active before any impact is detected. As soon as the first impact is detected, a switch is made to the interim mode, following the approach of [16]. The goal of this interim mode is to prevent control input peaks that can trigger vibrations, damage or destabilize the system and increase energy consumption in the presence of ideally simultaneous impacts. Uncertainties or disturbances can cause a time separation in the impacts between both robots and the object, which are planned to be executed simultaneously. So, in practice, there will always be a short time where the system is neither in the ante-impact state, nor in the post-impact state, meaning that neither reference can reliably be used for tracking control. Switching to the post-impact mode immediately or staying in the ante-impact mode can create the risk of inducing control input peaks. The interim mode solves this by initially removing velocity feedback and gradually increasing the velocity feedback control gain over time, while also gradually increasing the desired grasping force to promote contact completion. After a fixed time, when the impact event is assumed to be completed, a switch is made to the post-impact mode, where the post-impact velocity and force reference generated in 3.2 is followed.

On task-space QP control enhancement, the aim is to deal with impact without considerably changing its structure, i.e., envisioning handling impact tasks simply by adding or reformulating task objectives and constraints without introducing new decision variables. We build our QP improvement, see Figure 16,

[2] mc_rtc: https://jrl-umi3218.github.io/mc_rtc/

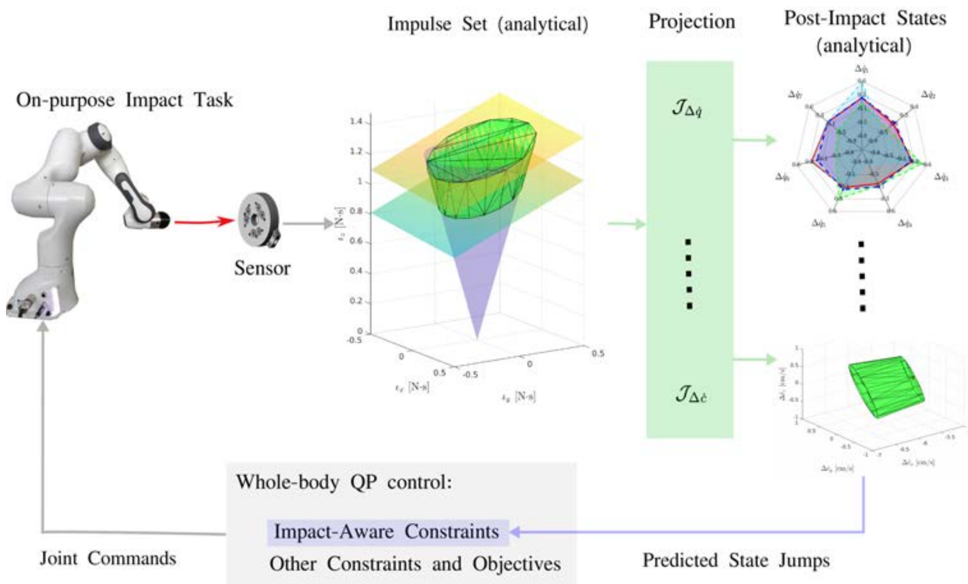


Figure 16: The impact-aware QP regulates the contact velocity in a modified search space to ensure that the post-impact state jumps are hardware-affordable.

on top of our initial concept proposed in simulation for fixed-base robots in [17] and on our experimental modeling work done in I.AM. and published in [18, 19]. The main results in I.AM. for this task are gathered in a journal paper (IJRR) [20].

3.4 Grab Contact State Sensing

This section describes the implementation details of the tasks constituting I.Sense (see Section 1).

3.4.1 Impact detection

The reference spreading technique relies on an impact detector to switch the desired velocity vector field (fed to the controller) at impact time. Therefore, it is crucial to have a sensitive impact detector that is able to detect the impact occurrence with minimal delay while at the same time guaranteeing robustness to false positives.

This has been achieved using the impact detection technique described in the appendix of [16]. The detector uses only proprioceptive sensing, namely the position encoders and torque sensors at the joints. The former are used to estimate the end effector velocity, while the latter are used to calculate the external force at the end effector via a momentum observer; in our case, both estimates are provided by the Franka Control Interface. Given an end effector velocity jump (i.e., abrupt change in velocity), the detector core insight to achieve both sensitivity and robustness is to trigger an impact detection only if a significant external force acts on the end effector in the same direction as the velocity jump.

3.4.2 Contact Force and Object Mass estimation

F/T sensors can be mounted between the robot flange and the end-effector to estimate the contact wrench at the end-effector accurately. However, adding additional sensors to the robot reduces its maximum payload and only senses external interaction at the end-effector. In our scenarios, we are using robots equipped with joint torque sensors in each joint. By utilizing the sensed torques and an accurate robot model, the contact wrench can be estimated using a given observer, such as the Momentum-Observer [21], without requiring any additional hardware.

However, for accurate robot modeling, the inertial parameters of the arm must be identified first. This identification is based on a linear matrix inequality approach [22], utilizing generalized robot base parameters [23]. With knowledge of the robot dynamics and kinematics, the external wrench can be estimated using the Momentum-Observer [21].

The fundamental principle underlying the model-based contact detection technique is based on estimating the manipulator's generalized momentum, denoted as $\boldsymbol{p}$, achieved through the generation and monitoring of a residual signal $\boldsymbol{r}_M$. Here, the individual components are filtered versions of the external joint torque $\boldsymbol{\tau}_{ext}$. In the subsequent discussion, we provide a brief overview of this disturbance observer and highlight its essential characteristics [21]. The dynamics of a generalized momentum observer for a rigid-joint robot manipulator are elaborated in [24].

$$\begin{aligned}\boldsymbol{p} &= \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}\\ \dot{\hat{\boldsymbol{p}}} &= \boldsymbol{\tau}_{\mathrm{j}} - \boldsymbol{\tau}_{\mathrm{f}} - \hat{\boldsymbol{\beta}}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{r}\\ \boldsymbol{\beta}(\boldsymbol{q},\dot{\boldsymbol{q}}) &:= \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \dot{\boldsymbol{M}}(\boldsymbol{q})\dot{\boldsymbol{q}}\\ \dot{\boldsymbol{r}}_M &= \boldsymbol{K}_O(\dot{\boldsymbol{p}} - \dot{\hat{\boldsymbol{p}}})\end{aligned} \tag{17}$$

where $\hat{\boldsymbol{\beta}}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is an auxiliary vector obtained from the available robot dynamics model, and $\boldsymbol{K}_O$ is the constant element-wise positive diagonal gain matrix. Solving for the momentum observer output gives [21].

$$\boldsymbol{r}_M(t) = \boldsymbol{K}_O\left(\boldsymbol{p}(t) - \int_{\tau=0}^{t}\dot{\hat{\boldsymbol{p}}}(\tau)\mathrm{d}\tau - \boldsymbol{p}(0)\right), \tag{18}$$

which implies the dynamics of the residual monitoring signal

$$\dot{\boldsymbol{r}}_M = \boldsymbol{K}_O\left(\boldsymbol{\tau}_{\mathrm{ext}} - \boldsymbol{r}_M\right) \tag{19}$$

The transfer function for each component is given in the Laplace domain by

$$r_{M,i} = \frac{k_{O,i}}{s + k_{O,i}}\tau_{\mathrm{ext},i} = \frac{1}{1 + T_{O,i}s}\tau_{\mathrm{ext},i}, i = 1,\ldots,n \tag{20}$$

with the component-wise time constants $T_{O,i} = \frac{1}{k_{O,i}}$ for transient response. A key characteristic of these residual filters can be obtained at the limit $\boldsymbol{K}_O \to \infty \Rightarrow \boldsymbol{r} \approx \boldsymbol{\tau}_{ext}$, which makes the momentum observer serve as a virtual sensor for external joint torques acting along the robot structure [21].

The external wrench $\boldsymbol{F}_{ext}$ can be estimated using the pseudo-inverse of the Jacobian $\boldsymbol{J}^{\#T}$

$$\boldsymbol{F}_{ext} = \boldsymbol{J}^{\#T}\boldsymbol{\tau}_{ext} \tag{21}$$

The mass of any grasped object can be identified in real-time by estimating the external wrench. When representing the external wrench in the robot base or world frame, the axis parallel to the gravitational acceleration vector indicates the gravitational force related to the object's mass. In an n-robot setup, the total estimated mass of the object m_{obj} is given by:

$$m_{obj} = \frac{\sum_{i=1}^{n} \langle \boldsymbol{a}_{grav}, \boldsymbol{F}_{ext,i} \rangle}{||\boldsymbol{a}_{grav}||^2} \tag{22}$$

where $\boldsymbol{a}_{grav}$ denotes the gravitational acceleration vector.

3.4.3 Failure detection and classification (for reflexes triggering)

The failure recovery strategy adopted is grounded on the framework introduced in [25], which tackles the problem of failure recovery as one of the selection of the appropriate recovery action for the failure that occurred. The recovery action (called *reflex*) is selected among a predefined set to allow a quick response. Indeed, using predefined recovery actions reduces the delay from failure detection to the start of recovery execution to only the time needed to select the recovery action—no extra computation burden is required for setting up the recovery action.

The reflex selection strategy is defined by the *Reflex selection function* r, which has its domain in the space of possible failures, called *Reflex context taxonomy*, and values in the space of recovery actions, called *Reflex control capsules*.

The *Reflex context taxonomy* must be defined by domain experts, individuating the most occurring and costly (i.e., if not recovered) faults. For our implementation, we considered two faults: having a parcel with a different mass than expected (usually happening in logistics if a quantity discount is not noted) and having a parcel with different dimensions than expected (this might be caused by a quantity discount as well, but also from a vision system error, if employed). The former fault is detected leveraging the payload estimation produced by the module described in 3.4.2. The latter fault is detected considering the end effector pose, comparing the nominal impact pose with the pose at impact time, which is detected using the module in 3.4.1,

Such a scheme requires forecasting problems to create ad hoc reflex actions to recover the failure. But, there might be problems that the field expert might not predict. For this reason, we employed a catch-all classifier that detects if the task execution diverges from the nominal one. The knowledge about the nominal execution consists of sensory inputs gathered data during the nominal execution of the depalletization task, allowing the creation of a task model used by the classifier.

In particular, the classifier utilizes a time-dependent statistical model of the Cartesian poses of specified end-effectors (in this experiment, the two robot arm end-effectors) and their corresponding contact wrenches, effectively encoding the intrinsic task dynamics for a known controller and manipulation strategy for a given manipulation scenario [26]. The model can incorporate possible variations in control, initial robot poses, and box variables such as weight, dimensions, or pose, theoretically resulting in a robust yet sensitive model for anomaly detection.

In essence, the model delineates a feasible subspace of states for the robotic manipulation system for the given tactile manipulation task, effectively defining the measurements for which task execution is assumed to be correct. The model is able to evaluate task execution in real time. In contrast to deep-learning approaches, our model is physically interpretable, computationally lightweight, and memory efficient. Furthermore, such models may be employed to distinguish between different instances that feature discernible dynamics, such as different boxes.

Such a pattern recognition method triggers unexpected behavior also in case the fault is individuated by one classifier in the discriminating layer. Therefore, its output is considered by the Reflex Engine only if no other classifier detects a fault, as shown in Fig. 17.

Each *Reflex control capsule* should be a recovery action general enough to be effective for every scenario where the fault can happen. We implemented two reflex control capsules: *reverse replay reflex* and *put down reflex*. The first, when activated, tracks the last executed trajectory backward. In particular, during the nominal task execution, the end effector position and velocity measured sensors and target wrench are stored in a `deque`. When the reflex is triggered, the deque is read backward, sending target variables to the `mc_rtc` controller. The underlying reason for such implementation is that the robot visits configurations[3] already explored before the fault. Therefore, it does not require information on the environment state from perception systems to avoid collision—this is reasonably true if the robot is the only actor in the environment changing its elements configuration.

The *put down reflex* is tailored for the fault caused by grabbing a box with mass different from the nominal one. This fault is detected while having the package lifted and grabbed by the bimanual robot setup—necessary given the mass estimator functioning. Notice that having a different mass than the nominal one makes the reverse replay reflex unsafe since the target forces applied might not be sufficient for avoiding slippage. For this reason, the put down reflex consists of applying a target force in the direction perpendicular to the box surface in order to have a firm grip and put down the object on the support below. The maneuver finishes when a vertical force higher than a predefined threshold is perceived, meaning that the box is in contact with the support surface, and the grab can be terminated.

The reflex function r adopted gives the *put down reflex* for the wrong box mass fault, and *reverse replay reflex* for the wrong box dimensions fault and the generic fault (detected by the catch-all classifier).

The reflex control capsules are implemented as `mc_rtc` states. When a classifier detects a fault, an immediate state change occurs from the current state execution to the reflex state. Each reflex has a termination condition that occurs when the recovery is completed.

Clearly, the proposed scheme can be populated with endless fault classifiers and reflex control capsules, so our realization, rather than being a finite product, aims to be an example validating the rationale underlying the scheme and guiding future field experts in implementing their own solutions tailored to their application needs.

[3]the control scheme used locks the first joint, making the robot nonredundant. Therefore, there is a unique correspondence between end effector pose and robot joint configuration

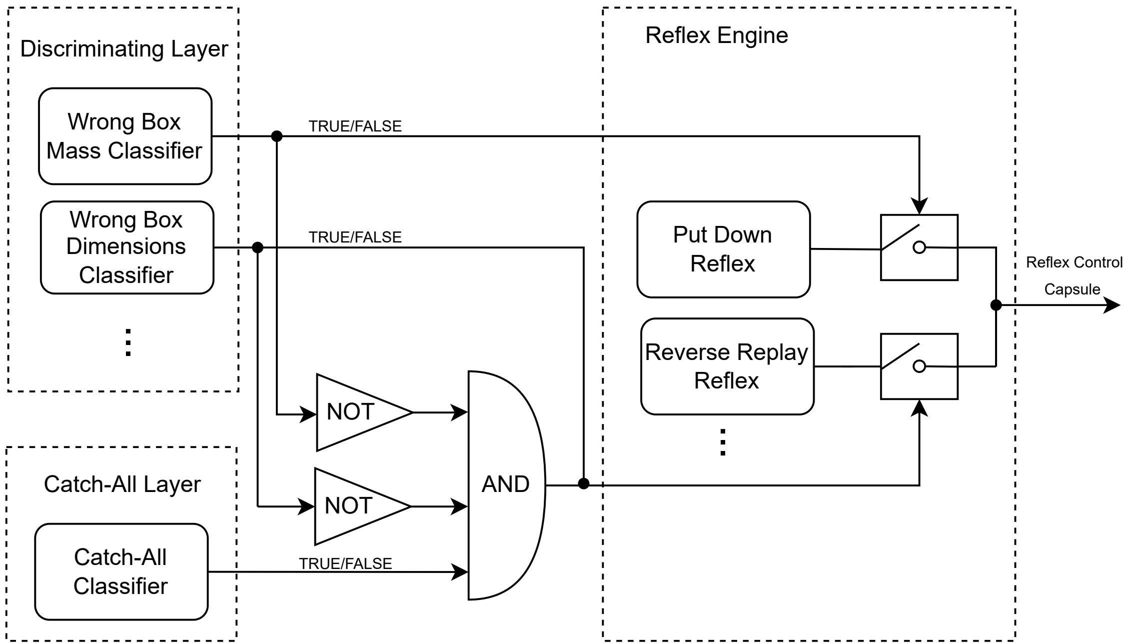


Figure 17: I.Sense fault detection and reaction pipeline. The classifiers output `TRUE` if a fault happens, and `FALSE` otherwise.

3.5 GRAB Simulation Environment

Over the past two decades, there has been a surge in the development of dynamics simulation software tailored for nonsmooth mechanical systems. These tools have emerged both within academic prototypes (such as Bullet, Open Dynamics Engine (ODE), and PhysX) and in commercial offerings (including Vortex, MuJoCo, and AGX Dynamics, the latter developed by Algoryx, a partner of I.AM.). Despite their widespread adoption, these simulation packages encounter various challenges when it comes to accurately computing contact geometries, impulses, and contact forces, particularly in scenarios involving friction.

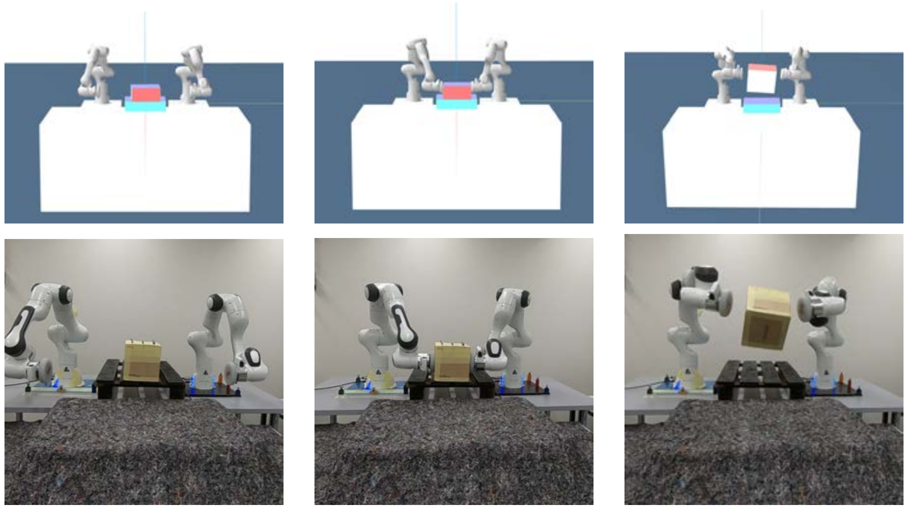
Nonsmooth mechanics shares resemblances with hybrid system theory, commonly applied in designing controllers for robotic systems subjected to impacts. However, in the realm of numerical simulations, the formalism of hybrid systems typically aligns with event-driven approaches, resulting in unpredictable computational workloads over fixed time intervals. This can lead to scenarios like Zeno behavior, particularly problematic in simulating motions involving impacts with multiple potential contact points and intricate geometries, as envisioned in projects like I.AM.

AGX Dynamics serves as a powerful tool for modeling and simulating complex mechanical systems in motion. It offers versatile applications for analysis, design, and serving as a physics engine for interactive 3D environments or virtual training scenarios for AI and ML applications. One of its key strengths lies in its ability to effectively handle impacts, contacts, and friction using nonsmooth numerical solvers, which operate at fixed intervals and aggregate impacts as simultaneous multi-way transitions.

In this integration effort, the complete pipeline for the pick-and-toss task was executed within AGX Dynamics. The motion generation employed the DS approach developed by EPFL, while the contact handling utilized the reference spreading technique from TU/e to manage impulses at contacts. Additionally, the low-level QP, facilitated by mc_rtc and developed by CNRS, controlled the robot behavior, ensuring adherence to joint position, velocity, and torque constraints.

In Figure 18, screenshots of the integrated AGX Dynamics simulation for the I.AM. integrated toss tasks are presented, with further evaluation discussed in the subsequent section. Within this simulation, the robot environment is initialized, and the robot proceeds to make contact with the box, lift it, and then toss it toward a predetermined target. This framework is advantageous as it operates using the same code utilized for real-world robot operation, enabling consistent measurement of outputs in both simulated and real-life scenarios. A representative plot illustrating this behavior is depicted in Figure 19. A comparison with data from real-world observations, as presented later, reveals that the impact event results in significantly higher force in simulation. This discrepancy is expected, as the simulated box is modeled as perfectly rigid, while real-world experiments demonstrate compliance in the box, resulting in lower peak contact force. Nonetheless, the overall behavior of the controller and timing remains comparable, underscoring the value of this simulation framework in the development of the control architecture. Given this tool, we can safely use AGX Dynamics to test the controller in a safe environment, which is especially useful given the potentially damaging nature of experiments with impacts. On top of this, AGX Dynamics can be used for initial control parameter tuning, before refining the parameters using experiments with the actual system.

Lastly, as discussed in Section 3.2.2 AGX Dynamics is used in the reference spreading methodology to estimate the velocity jump for a range of ante-impact configurations. The aim of this so-called impact map is to match the post-impact velocity reference with the predicted velocity directly after the impact, to remove peaks and jumps in the velocity tracking error, which otherwise result in undesired jumps in the input signals. To do so, the batch simulation feature, developed throughout the project as part of the GLUE framework, is used to initialize the system in a range of possible ante-impact states. Simulations in each state are then performed with pre-defined ante-impact velocities for a short time, capturing the impact event, and the resulting post-impact velocities are written to an HDF5 file. Through interpolation, the desired velocity corresponding to the impact state is determined as soon as the impact is detected. The result is a reduction of input peaks and failed grasps in the Benchmarking Experiments presented in Section 4.



(a) Initial position (b) Grabbing of the box Tossing of the box

Figure 18: AGX simulations for the toss task: (top row) Screenshots of the behavior of the robot in an AGX Dynamics simulation (bottom row) picture of real robot experiments performed in the lab at EFPL. (a) robots and box initial position, (b) the contact event where the robot arms grab the box, and (c) a screenshot after the release time demonstrates the box flying in the air. Note: the hardware methods and result are presented in detail in the following section.

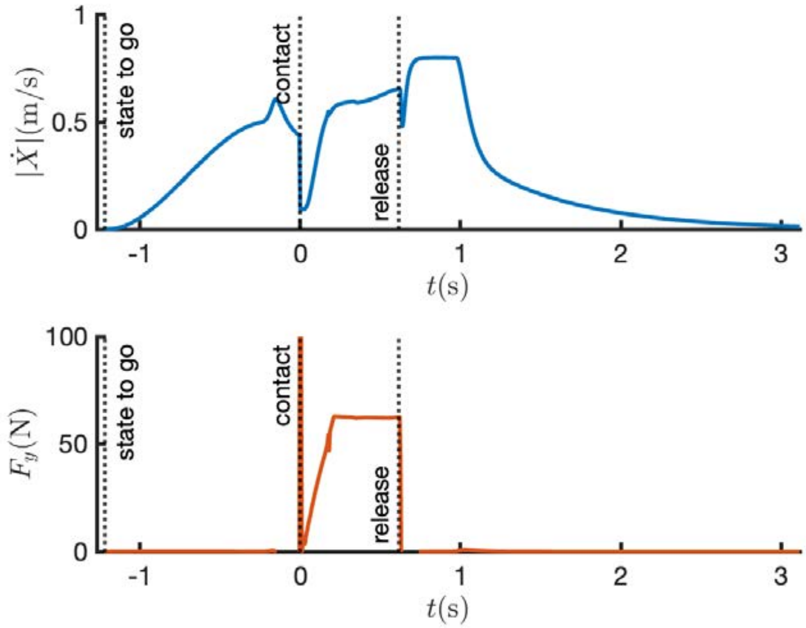


Figure 19: Qualitative plots depicting the norm velocity and normal force dynamics during the simulated toss task. A single trial with labeled events is depicted, serving as a representative example featuring an impact velocity of 0.4 m/s, a lift velocity of 0.6 m/s, and a box mass of 0.5 kg. (Top row) End-Effector Speed: The norm velocity of the robot's end-effector is depicted, representing the speed of the end-effector. (Bottom row) Contact Force Dynamics: Force exerted in the direction of contact with the box during the task is illustrated. Time zero corresponds to the impact moment, marked by the increase in normal force. Note the force plot at contact exceeded a value of 100 N but is cut off for the sake of visualization.

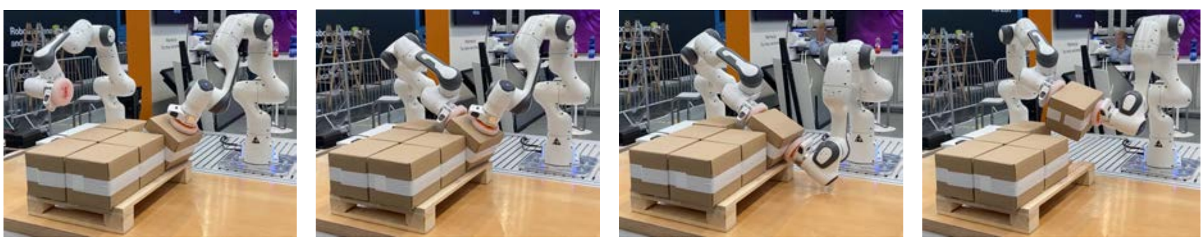
4 Benchmark of I.AM. GRAB Scenario against industry state of practice

This section is divided into two parts. First, we introduce the methods by which we systematically assess the performance of the GRAB scenario and compare it with current standard technology. Then, we present the results of our benchmarking experiments.

4.1 Methods

The section starts by describing the test scenarios and procedures, the test setups, the selected set of manipulated items, and the characteristics of the considered tasks. Next, it describes the evaluation metrics or key performance indicators (KPIs) used to asses and benchmark the GRAB scenario. Finally, it presents the methods for robustness testing.

While motion planning methods for the grasping and hitting of single objects have been developed throughout the project, as highlighted in Section 3.1.1 and 3.1.2, no motion planning module for autonomous grasping of objects from a tightly packed pallet was developed. Given the necessity of such a module to perform depalletization, research into this topic will be continued after the project has concluded. We have, however, shown that this task can be performed with a dual-arm robot setup, using custom silicon end effectors developed throughout the project. To achieve this, a teleoperation scheme has been developed throughout the project, using two HTC VIVE controllers and the mc_rtc control framework. As depicted in Figure 20, this scheme can be used to execute a tilt-and-grab strategy to successfully remove a single object from a tightly packed pallet.



(a) Tilt box from top surface (b) Establish contact on side surface with single robot (c) Establish contact on side surfaces with both robots (d) Lift box

Figure 20: Snapshots of the tilt and grab procedure of parcels in a densely stacked pallet, performed through teleoperation, and shown at the Automatica fair in 2023.

However, given the ongoing development of the autonomous planner to perform the motion shown in Figure 20, the validation of the GRAB scenario in the remainder of this section will focus on the grasping of single parcels with nonzero velocity. For this validation, the motion generation and force generation from Section 3.2.1 (I.Learn), the reference adaptation for impacts from Section 3.2.2 (I.Model), the impact-aware controller from Section 3.3 (I.Control), and contact state sensing from Section 3.4.1 (I.Sense) were integrated in hardware at EPFL.

To validate the GRAB scenario, we will introduce three tasks. The first task, known as the **pick and lift** task, involves the robot moving into contact with the box and lifting it to a specified height. This task allows us to isolate the impact event effectively. It will be used to specifically study the ability of the integrated system to make impulsive contact prior to assessing the entire system performance. The second task, referred to as the **pick and place** task, requires the robot to move into contact with the box, raise it, and then set it down on a pallet. Lastly, in the **pick and toss** task, the robot moves into contact with the box, lifts it, and then tosses it to a desired position on the pallet.

Robotic state-of-the-art Grabbing

As discussed in Section 3, the current state of the art in robotic manipulation is to make and break contact quasi-statically – such that velocity is negligible. Thus, in this experiment, for making contact, the speed of impact condition included a value with zero impact velocity. Furthermore, for breaking contact, the place and toss task allowed us to compare the benefits of releasing with non-zero velocity – tossing.

Robotic I.AM. Grabbing

The key contribution of the work is the grabbing in which we demonstrate the speed and effort benefits of using impact-aware manipulation to safely decrease the cycle time by making impulsive contact with the box and tossing it.

4.1.1 Experimental Setup

The two setups used in the GRAB scenario are described in this chapter. As the integration focused on the dual panda setup, we selected it to perform the validation of the scenario.

Setups with Dual arm Franka Emika robots

In this setup, two Franka Emika robots are placed side by side. A TU/e soft-pad is attached to the end-effector of each robot. The robots can grab a box located on the pallet between them. If the box is then placed or tossed, it will land on the plastic pallet placed in front of the robots. An OptiTrack system is installed to track the position of both robots (to allow the setup to be robust to changes in robot positions) and the box. Data was recorded at 1000 Hz.

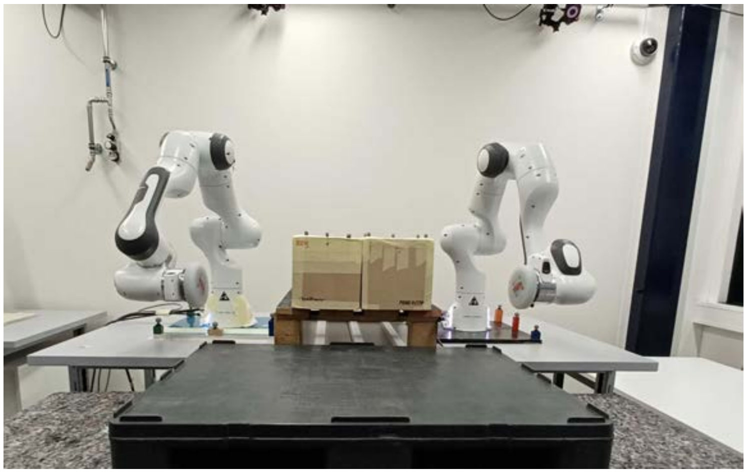


Figure 21: Setup for dual-arm grabbing with FRANKA EMIKA robots

Setups with Dual arm KUKA LBR IIWA robots

A KUKA IIWA 7 is placed next to KUKA IIWA 14. An EPFL soft-pad is attached to the end-effector of each robot. The robots can grab a box located on the pallet between them. An OptiTrack system is installed to track the position of both robots (to allow the setup to be robust to changes in robot positions) and the box.

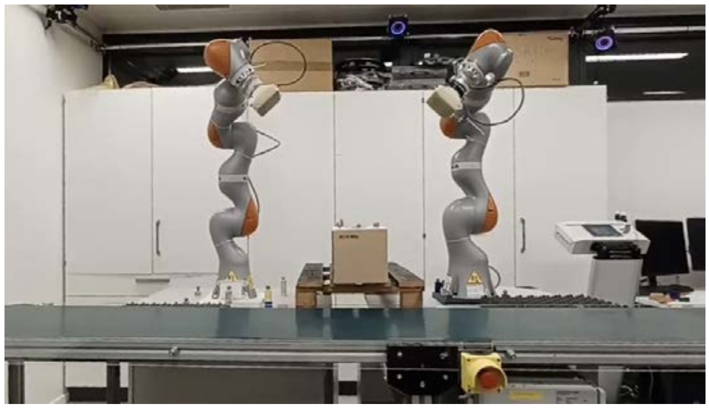


Figure 22: Setup for dual-arm grabbing with KUKA IIWA 7 and IIWA14 robots

Items

In order to show the robustness of the grab scenario, three boxes of different weights and dimensions were selected. The first box was filled with foam and rice, it weighed 0.5 kg, and its dimensions were $[0.190, 0.183, 0.189]$ m. The second box was filled with cardboard, it weighed 1.0 kg, and its dimensions were $[0.187, 0.289, 0.185]$ m. Finally, the third box was filled with fabric, it weighed 1.5 kg, and its dimensions were $[0.268, 0.360, 0.248]$ m. These boxes were selected to be representative of the types of boxes which would be observed in an industry setting.

(a) 0.5 kg box (b) 1.0 kg box 1.5 kg box

Figure 23: The different boxes used for the experiments. The boxes were intentionally selected to have different masses, shapes, and internal properties. The boxes were generally full such that the content did not move around. The 0.5 kg box was filled with foam and rice, the 1.0 kg box was filled with cardboard, and the 1.5 kg was filled with fabric.

4.1.2 Tasks

The tasks are defined in detail below. For each task, two types of speed parameters were investigated to benchmark the system performance. The first speed parameter was the **impact speed**, this is the desired speed of impact used in the DS controller. Second, was the **task speed**, this was the desired speed of the DS to perform the lift, place, or toss after making contact with the box. For each combination of conditions (task, box, impact speed, and task speed), the task was performed for 20 trials. This resulted in 1,680 trials. The experimental conditions and parameters are summarized in Table 1.

Condition	Number of levels	Parameters
Task	3	Lift, Toss, and Grab
Box	3	0.5 kg, 1 kg, and 1.5 kg
Impact Speed	4	0.0, 0.2, 0.3, and 0.4 m/s
Task Speed	2-3	Lift: 0.4, 0.6, and 0.8 m/s Toss and Place: 0.4 and 0.5 m/s

Table 1: Summary of the task conditions and parameters.

For each condition, the robot first performed the Pick operation, which started by sending a go command, at this time both pandas started to follow the DS described in Section 3.2.1 toward the box [27, 12]. As the robots approached the box, the reference spreading approach described in Section 3.2.2 ensures the avoidance of impact-induced jumps and peaks in the control inputs, reducing the risk of failure during and after the impact sequence [16]. One of the parameters that varied in the experiments was the desired impact speed. This represents the changing of the parameters of the DS used to control the robot prior to contact. The desired impact speed was varied from 0.0, 0.2, 0.3, and 0.4 m/s. Following contact, the box was raised. This is where the difference between the three tasks begins.

Lift

Following the pick, the lift task commenced, involving raising the box to a target height of 0.45 m and maintaining it in a stationary position. This condition was examined to focus on the pick operation, where impact may occur upon contact with the box. By isolating this condition, we eliminated additional complexities present in subsequent pick and toss conditions. The lift task was executed at four distinct desired speeds: 0.4, 0.6, and 0.8 m/s. This process is illustrated in Figure 24.

The industry standard condition for the pick and lift task involves making quasi-static contact, corresponding to an impact velocity of 0.0 m/s. In contrast, the I.AM. condition represents the fastest impact speed tested, which in this case was an impact velocity of 0.4 m/s.

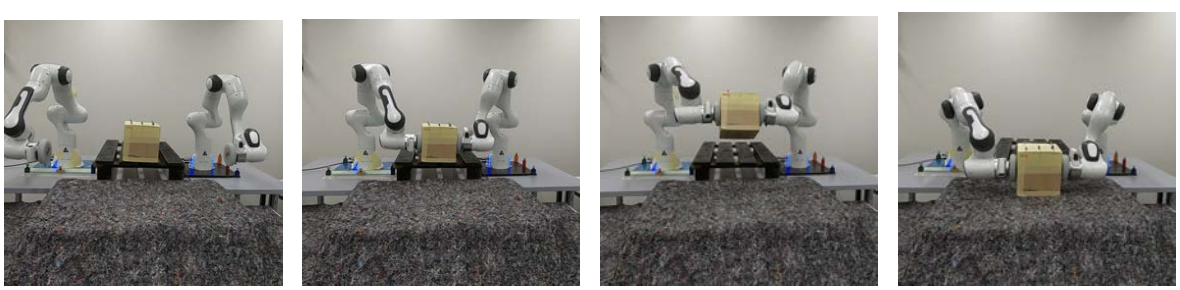
(a) Initial position (b) Grabbing of the box (c) Lift final position

Figure 24: Lift Experiments: (a) robots and box initial position, (b) the contact event where the robot arms grab the box, and (c) final lift position where the robot holds the box statically at the desired target position. Note the key difference between this Figure and Figure 25 is pannel (d) where the box is placed quasi-statically instead of tossed.

Place

After being picked the place task began. In the place task, the box was first contacted with a desired impact speed, then lifted and placed at a target position. It is important to note that the placing action is quasi-static. Figure 25 illustrates this process. After the box was released the robot was returned to its starting position and remained there with zero velocity ready for the next trial.

The industry standard aims to pick and place items as quickly as possible while maintaining quasi-static conditions. Therefore, a subset of this task condition, the case with zero impact speed and maximum task speed, represents the benchmark condition and involves the robot making contact with zero impact velocity and placing the box as swiftly as possible. This represents the industry standard for minimizing contact time while remaining quasi-static. In the later part of the document this condition and parameter range will be denoted the **industry standard** condition.



(a) Initial position (b) Grabbing of the box (c) Lifting the box (d) Placing the box at its final position

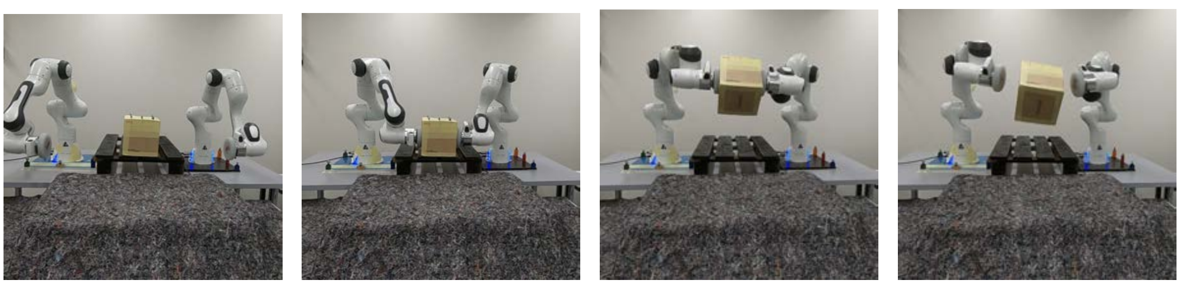
Figure 25: Place Experiments: (a) robots and box initial position, (b) the contact event where the robot arms grab the box, (c) picture during the process of lifting the box forward toward the target position, and (d) picture when the robot places the object at the desired position. Note the control for the pick and toss conditions are the same except for the last frame (d).

Toss

Finally, the faster version of the place task is the toss task – this is extension of the Grab scenario defined in the consortium agreement. In this case, the box is picked in the same manner as the other tasks. However, in this case the robot does not move all the way to the goal to set down the box quasi-statically. Instead, the robots followed a DS to approach a desired release position and velocity such that the box landed at a desired position.

The toss task represents the faster iteration of the place task. Similar to the lift and place tasks, the box is picked up using the same method. However, rather than moving all the way to the goal to place the box quasi-statically – as was done for place. In the toss task, the robot follows a trajectory generated by a DS to approach a desired release position and velocity. When this desired state is approached the robot releases the box such that the box lands in the desired position.

A specific parameter range that is important to note for this condition is the fastest impact speed and fastest toss speed which could be achieved – this was denoted the **I.AM. Grab** condition. This is the condition that was compared to the industry standard. Figure 26 illustrates this process.



(a) Initial position (b) Grabbing of the box (c) Lifting the box (d) Tossing of the box

Figure 26: Toss Experiments: (a) robots and box initial position, (b) the contact event where the robot arms grab the box, (c) picture during the process of lifting the box forward toward the target position, and (d) picture when the robot has released the box which was flying toward the goal. Note the control for the pick and toss conditions are the same except for the last frame (d).

4.1.3 Performance Evaluation: Metrics (KPIs)

To assess the performance of the integrated I.AM. controller for a dual-arm robot, we compare it with state-of-the-art dual-arm pick-and-place technology characterized by near-zero relative contact velocities. Additionally, we conduct a sensitivity analysis at both system and task levels to evaluate the contribution of specific components to overall system performance. This analysis also investigates the impact of changes in object mass and position on task performance. This evaluation includes grabbing with impacts, and tossing/placing objects out of the pallet.

As metrics of evaluation, we consider the following key performance indicators (KPIs), namely:

- the average pre-impact time [s] (average time it takes to go from a distance of 10 cm from the box to the moment of contact);
- the average task time [s] (the average time it takes from grab to place);
- the average cycle time [s] (the average time it takes from one grab to the next grab).

- the average robot energy consumption per cycle [J] (for the comparison with the classical approach, we also consider the mechanical energy required by the tasks.)

Methodologically each of these parameters was computed based on estimates and thresholds. One measure important for detecting these states was the estimate of the time when the robot made or broke contact.

The **pre-impact time** was only computed for the pick and lift task, as it was designed to study the effects of making impulsive contact. It was defined as the time when the robot end-effector was within 10 cm of making contact with the box until contact with the box was detected. To explicitly compare the industry standard condition pick-and-lift to the I.AM. condition pick-and-lift the percent difference was computed between the conditions. The percent difference was computed by,

$$\text{Percent Difference Lift Time} = 100\left(\frac{\Delta t_{industry} - \Delta t_{I.AM.}}{\Delta t_{industry}}\right). \tag{23}$$

Where the $\Delta t_{industry}$ denotes the time of the industry standard condition, and $\Delta t_{I.AM.}$ denotes the time of the I.AM. condition.

The **pick and place time** as well as the **pick and toss time** were defined as the time from when contact was detected to the time the release was detected. For each condition the percent difference between the pick-and-place and the pick-and-toss was computed as,

$$\text{Percent Difference Toss Time} = 100\left(\frac{\Delta t_{Pick} - \Delta t_{Toss}}{\Delta t_{Pick}}\right). \tag{24}$$

Where the Δt_{Pick} denotes the time of the pick-and-place condition, and Δt_{Toss} denotes the time of the pick-and-toss condition. This provided a metric of how much faster pick-and-toss was than pick-and-place. In addition in a similar way the percent difference between the industry standard condition and the I.AM. condition was computed for these metrics. Such that the fastest pick-and-place was compared to the fastest pick-and-toss.

The **average cycle time** also varied depending on the task. For the lift task, the cycle time was not defined, as this task focused solely on studying the making contact portion of the behavior. However, for the place and toss tasks, the cycle time was defined as the duration from when the go command was sent to when the robot's norm velocity dropped below a threshold of 0.5 m/s. In the same way as done above the percent difference in average cycle time was computed to compare the pick-and-place to the pick-and-toss for each condition. The percent difference was also computed between the industry standard and the I.AM. conditions as done above.

The primary rationale for defining the pick and toss time is practicality: our main focus is on determining when the robot is ready to perform another task, rather than solely on when the object reaches its intended destination. This consideration ensures that our assessment aligns more closely with real-world operational needs and efficiency.

The **robot energy consumption** was defined as E, and computed by,

$$E = \Delta t \sum_{i=n}^{N} |\boldsymbol{\tau}_i^T||\dot{\boldsymbol{q}}_i| \tag{25}$$

where the vertical bars denote the absolute value, n denotes the time sample index which corresponded to the sample time where the hold command was sent, N denotes the time sample index which corresponded to the sample time where the box was released, the joint torque measured by the robot was denoted by τ, where the constant time duration between samples was denoted by Δt, and the joint velocity of the robot was denoted by $\dot{q}$. This metric was only defined for the place and toss condition. In the same way as done above the percent difference in average robot energy consumption was computed to compare the pick-and-place to the pick-and-toss for each condition. Then the percent difference was also computed between the industry standard and the I.AM. conditions as done above.

4.1.4 Robustness Experiments

For the robustness experiments the toss task in the I.AM. condition were tested on a broader selection of boxes. Firstly, a box containing liquid juice containers was utilized to simulate a typical box found in local grocery stores. Secondly, a box housing a loose drill chuck and water bottle was employed to underscore the significant mass of the object and its capacity for unrestricted movement within the box. Finally, the system's ability to withstand compressive forces throughout the task was assessed by positioning two boxes adjacent to each other. This configuration allowed for the simultaneous grasping of both boxes, demonstrating the system's robustness under such conditions. This object can be seen in Figure 27.

(a) 2.0 kg box (b) 1.6 kg moving objects (c) two boxes 0.6 kg

Figure 27: Boxes used in robustness experiments: (a) A box containing liquid juice containers. (b) A box containing a drill chuck and water bottle. (c) Two boxes positioned adjacent to each other.

4.2 Results

This section provides the results derived from benchmarking experiments conducted at EPFL. The presentation is structured into distinct segments to facilitate clarity and comprehension. Firstly, the lift results are presented this is an evaluation of impulsive contact in isolation from other behaviors. This includes an analysis of qualitative behavior, alongside Key Performance Indicators (KPIs) pertinent to this aspect. Following this initial assessment, the section proceeds to present qualitative plots pertaining to the place and toss tasks. These visual representations offer insights into the observed behaviors within these tasks. Subsequently, attention turns to the KPIs associated with the combined place and toss task. By juxtaposing these metrics, a comparative analysis between the two conditions is facilitated.

Lastly, the section culminates with a presentation of results highlighting the robustness of the employed methodologies. This serves to underscore the reliability and efficacy of the experimental approaches employed.

4.2.1 Lift

The lifting task was performed to evaluate the consequences of establishing contact with the box. Within this task, the robots engaged with the box, subsequently elevating it to a predetermined position. Figure 28 illustrates plots derived from both individual trials and trial averages, providing a visual representation of the observed behavior. Notably, an evident peak in the estimated contact force is discernible at the moment of impact. This is followed by a gradual increase of the contact force, as expected with the interim mode of the RS approach highlighted in Section 3.3, which ensures a lack of peaks in the input signals and a successful contact transition. After this interim mode is finished, a plateau in the contact force is observed as the box is raised.

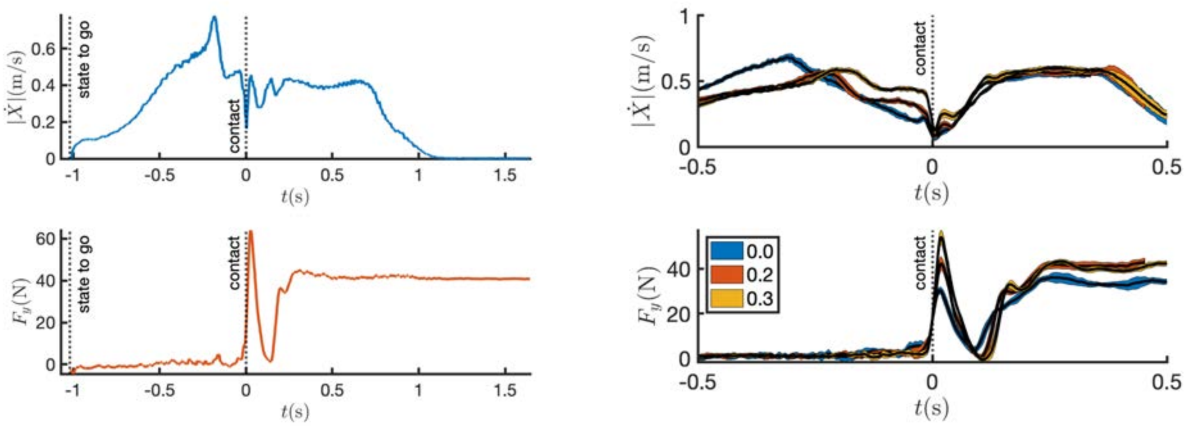


Figure 28: Qualitative plots depicting the norm velocity and normal force dynamics during the lifting task. These plots showcase representative trials involving a box mass of 0.5 kg. (left column) Single Trial Representation: A single trial with labeled events is depicted, serving as a representative example featuring an impact velocity of 0.3 m/s and a lift velocity of 0.4 m/s. (right column) Multiple Trial Averages: A plot displaying averages across multiple trials for varying impact speeds is presented. The black lines denote the mean values, while the shading indicates the standard deviation. The visualization highlights the impact of increasing desired impact speeds when the desired lift speed remains constant at 0.5 m/s. (Top row) End-Effector Speed: The norm velocity of the robot's end-effector is depicted, representing the speed of the end-effector. (Bottom row) Contact Force Dynamics: Force exerted in the direction of contact with the box during the task is illustrated. Time zero corresponds to the impact moment, marked by the increase in normal force. The primary reason for presenting the trial average plots in the right column is to illustrate the repeatability of the behavior.

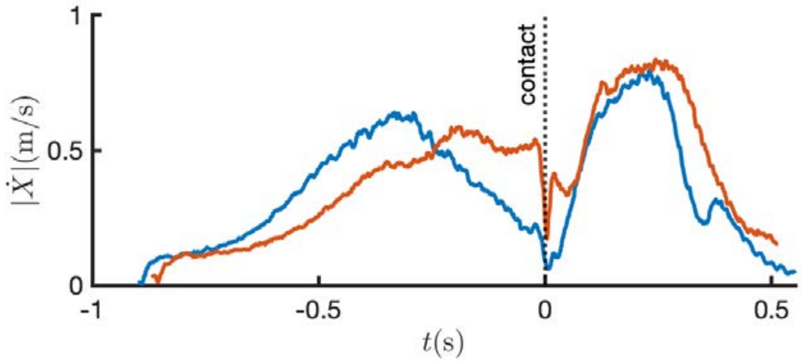


Figure 29: Velocity for the lift (blue) industry standard condition and the (red) I.AM. condition. Note that the industry standard condition must go slower to make quasi-static contact with the box while the I.AM. condition can safely make contact with a nonzero velocity.

Lift KPI

Here we present the pre-impact time. The key to this metric was to quantify the time that could be saved by being able to move into contact with the box with a nonzero velocity. We observed substantial decreases in the percent difference between the industry standard condition and the I.AM. condition – See Table 2 and Figure 29. However, it is important to note that this difference is on the order of 100 ms. While this is a short duration, this could make a large difference over many cycles.

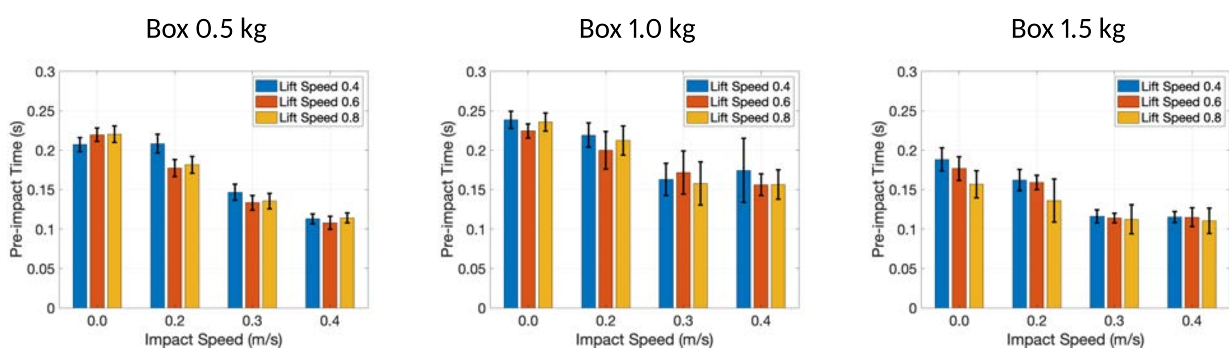


Figure 30: Average pre-impact time for each of the three boxes (left) 0.5 kg, (middle) 1.0 kg, and (right) 1.5 kg. The bars are grouped by impact speed, within a group of bars the color denotes the different task speeds according to the legend. The error bars indicate plus or minus one standard deviation from the mean.

	Box Type		
	0.5 (kg)	1.0 (kg)	1.5 (kg)
Percent difference in the pre-impact time	48%	34%	30%

Table 2: Summary of the percent difference in the pre-impact time between the industry standard condition and the I.AM. condition. Note that positive differences correspond to the case where the I.AM. condition out performed the industry standard condition.

4.2.2 Place

The place task was performed as a baseline to determine the benefit of tossing the box. In Figure 31 plots from the behavior for a single trial and trial averages are presented. It is clear that at the time of impact, there is a jump in force which then converges to a plateau while the box is lifted. Consequently the box is released and the velocity returns to zero again as the robot returns to its home position.

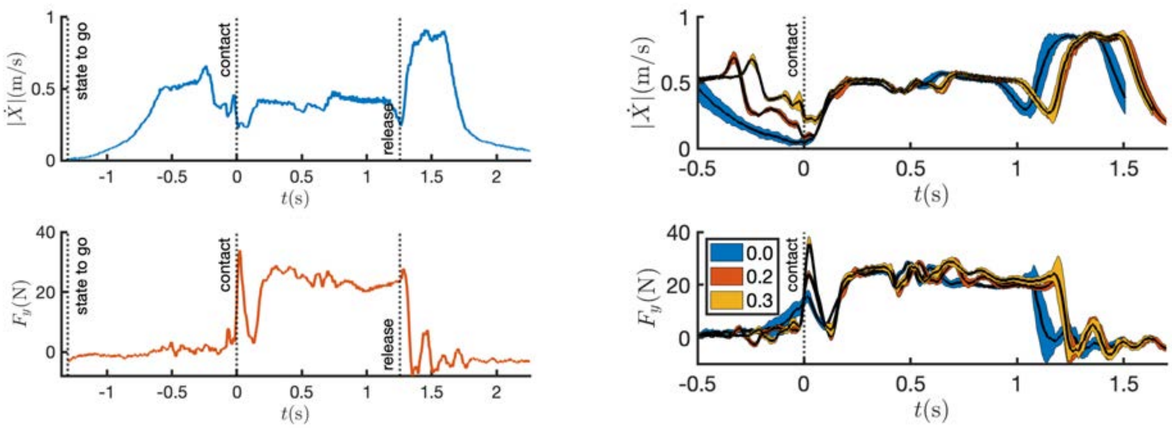


Figure 31: Qualitative plots depicting the norm velocity and normal force during the place task. These plots showcase representative trials involving a box mass of 0.5 kg. (left column) Single Trial Representation: A single trial with labeled events is depicted, serving as a representative example featuring an impact velocity of 0.3 m/s and a lift velocity of 0.4 m/s. (right column) Multiple Trial Averages: A plot displaying averages across multiple trials for varying impact speeds is presented. The black lines denote the mean values, while the shading indicates the standard deviation. The visualization highlights the impact of increasing desired impact speeds when the desired place speed remains constant at 0.5 m/s. (Top row) End-Effector Speed: The norm velocity of the robot's end-effector is depicted, representing the speed of the end-effector. (Bottom row) Contact Force Dynamics: Force exerted in the direction of contact with the box during the task is illustrated. Time zero corresponds to the contact time, marked by the increase in normal force.

4.2.3 Toss

The toss task was performed to assess the performance of the I.AM. Grab technology. In Figure 32 plots from the behavior for a single trial and trial averages are presented. This plot demonstrates a similar pattern to that of the place. However, it is important to note that the velocity does not decrease before release. In addition, the time it takes to release is far less as the robot end-effectors do not have to travel all the way to the target.

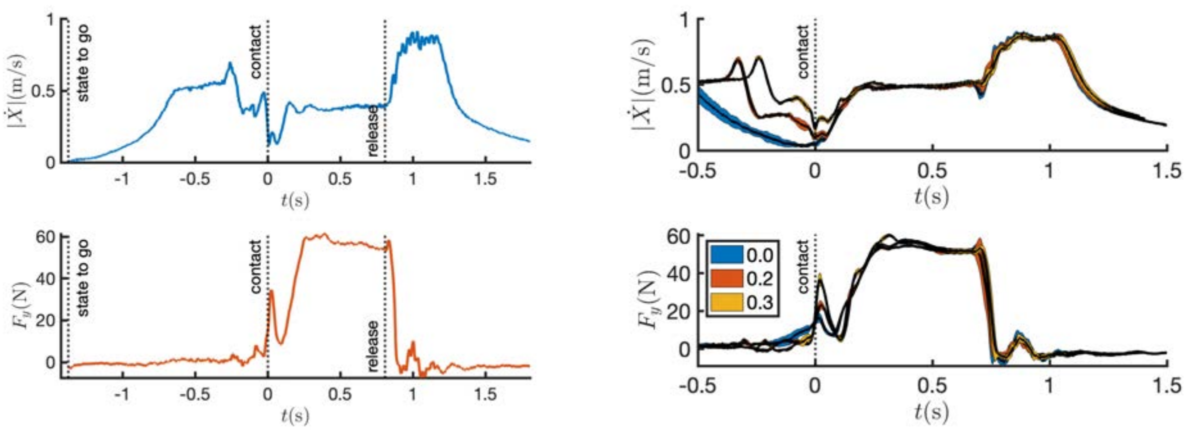


Figure 32: Qualitative plots depicting the norm velocity and normal force during the toss task. This plot uses the same structure as that of Figure 31. These plots showcase representative trials involving a box mass of 0.5kg. (left column) The single trial presented on the left features an impact velocity of 0.4 m/s and a lift velocity of 0.5 m/s. (right column) Multiple trial averages with a desired toss speed of 0.5 m/s.

4.2.4 Place and Toss Combined KPI Evaluation and Comparison

The goal of the KPIs is to quantify the performance of the combined system and assess its efficacy. Below we present the average task time (pick and place or pick and toss), the average cycle time, and the average Robot Energy. In Figure 34 (right) the bar represents the percent difference between the Pick and Place time and the Pick and Toss time. A positive value indicates that the toss task was faster. In fact, it is clear that the toss task was more than 30% faster than the comparable pick and place task. In Figure 35 (right) improvements were also observed for the average cycle time in all conditions. Finally, the toss task was more energy efficient than the place task, see Figure 36 (right), with a decrease in energy in all conditions. To avoid clutter in this section, the main text presents only the KPIs for the 1.5 kg box. However, for the interested reader, we present the same KPIs for the 0.5 kg and 1.0 kg boxes in Appendix A. Furthermore, this appendix also reports in the mean and standard deviation of each KPI in each combination of condition.

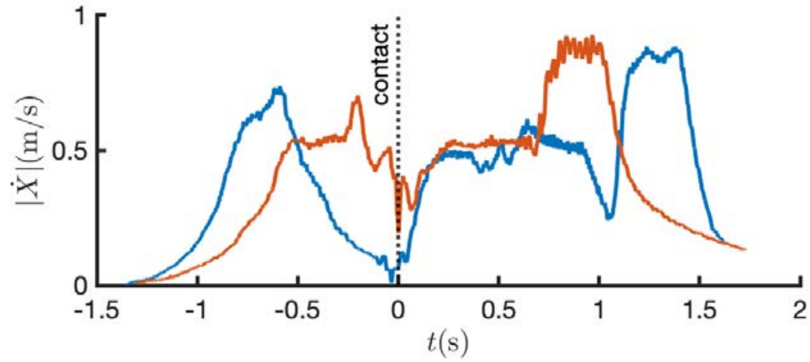


Figure 33: Velocity for the (blue) quasi-static pick and place – industry standard condition and the (red) impulsive pick and toss – I.AM. condition toss. Note that the industry standard condition must go slower to make quasi-static contact with the box while the I.AM. condition can safely make contact with a nonzero velocity. Furthermore, there is a large dip in the velocity for the industry standard condition around 1 second, this is required to place the box. While the toss task does not need to slow down during this time interval.

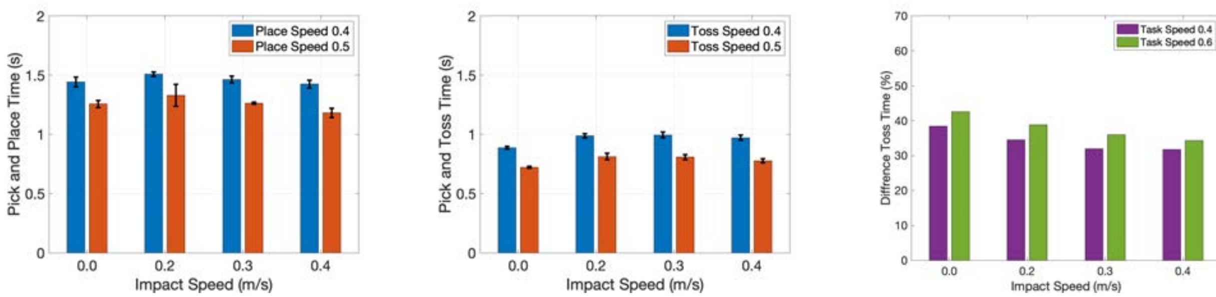


Figure 34: Average Task Time (Box 1.5 kg): (left) Pick and place task times, (middle) pick and toss task times, (right) percent difference between the pick and place vs the pick and toss task times. Note that a positive percent difference indicates how much faster the toss task was than the pick task. The bars are grouped by impact speed, within a group of bars the color denotes the different task speeds according to the legend. The error bars indicated plus or minus one standard deviation from the mean.

The key question is whether the industry standard condition (place: impact velocity 0.0 m/s, task speed 0.5 m/s) is better than the I.AM. condition (toss: impact speed 0.4 m/s task speed 0.5 m/s). In all box conditions, we observed a decrease of more than 34% in average task time, 8% in average cycle time, and 32% in average robot energy. The results for each box are presented in Table 3.

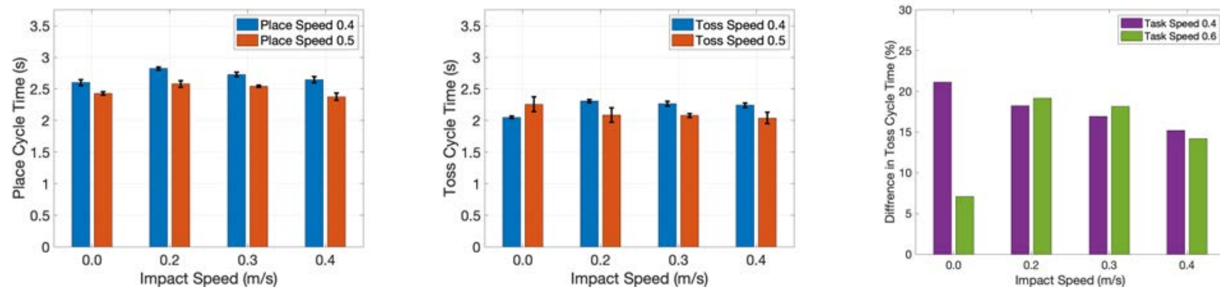


Figure 35: Average Cycle Time (Box 1.5 kg): (left) Place cycle time times, (middle) toss cycle times, (right) percent difference between the pick and place vs the pick and toss cycle times. The bar colors and organization are consistent with that of Figure 34.

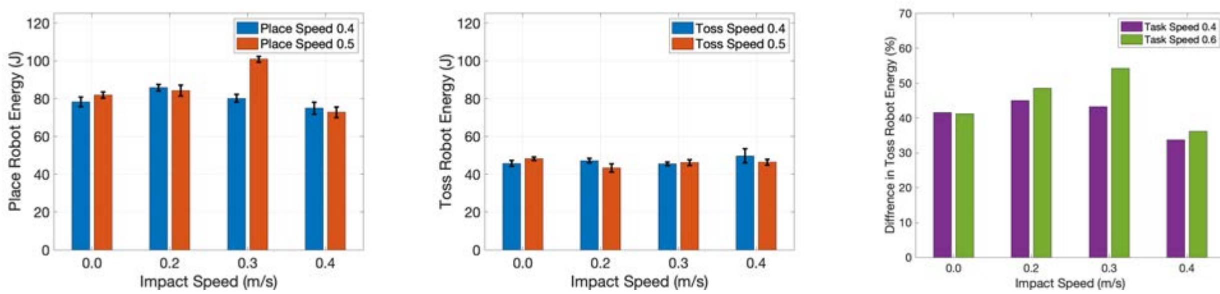


Figure 36: Average Robot Energy (Box 1.5 kg): (left) Place robot energy, (middle) toss robot energy, (right) percent difference between the place and the toss energy. The bar colors and organization are consistent with that of Figure 34.

	Box Type		
Average KPI	0.5 (kg)	1.0 (kg)	1.5 (kg)
Task Time	44%	42%	34%
Cycle Time	8%	13%	14%
Robot Energy	32%	39%	36%

Table 3: Summary of the percent difference in each KPI between the industry standard condition (place: impact velocity 0.0 m/s, task speed 0.5 m/s) and the I.AM. condition (toss: impact speed 0.4 m/s task speed 0.5 m/s). This table includes results from the different boxes presented in detail in the appendix. Note that positive differences correspond to the case where the I.AM. condition outperformed the industry standard condition.

4.2.5 Sensitivity Analysis (Robustness Evaluation)

To evaluate the robustness of the I.AM. Grab integrated system experiments were conducted using a broader selection of boxes. These test included, a box containing liquid juice containers, a box housing a loose drill chuck and water bottle, and simultaneous grasping two both boxes. The system was able to successfully pick and toss each of the three objects with out any failure cases. Pictures of these objects being grasped is presented in Figure 37.

(a) 2.0 kg box (b) 1.6 kg moving objects (c) two boxes 0.6 kg

Figure 37: Dual arm system picking and tossing the boxes investigated in the robustness experiments. (a) A box containing liquid juice containers. (b) A box containing a drill chuck and water bottle. (c) Two boxes positioned adjacent to each other.

4.2.6 Interpretation of the Results

In this study, we present the contributions of the I.AM. project, encompassing the integrated components of I.Model, I.Learn, I.Sense, and I.Control, within a unified platform. The primary objective of the I.AM. project is to advance robotics technology, particularly in the context of logistics operations, by introducing impact-aware technologies.

Key Results

Our platform demonstrates notable advancements, particularly in achieving faster task completion and improved energy efficiency compared to conventional industry methods reliant on quasi-static contact.

For example, in the case of the 0.5 kg box, our toss task showcased a 44% reduction in average task time compared to the industry standard place task. This substantial efficiency gain highlights the potential of impulsive contact strategies to significantly minimize task duration and enhance the speed of logistics operations. Furthermore, analysis of average robot energy consumption revealed a 32% decrease in energy expenditure during the toss task compared to the place task, further emphasizing the benefits of impact-aware technologies in optimizing resource utilization without compromising task performance.

Another critical aspect worth highlighting is the substantial time savings achieved through the tossing action compared to the quasi-static pick action. In our experiments, the tossing action saved approximately 0.5 seconds per cycle, significantly outperforming the impulse pick action, which saved around 0.1 seconds per cycle. While both strategies contribute to reducing task time, the magnitude of time saved by tossing is notably higher, especially in scenarios where the task design permits such action.

This disparity in time savings underscores the pivotal role of the tossing action in expediting logistics operations, particularly in tasks involving repetitive actions and large volumes of objects. For tasks requiring thousands or even millions of cycles, the cumulative time savings achieved through tossing or impulsively picking can be substantial, leading to significant improvements in overall operational efficiency and throughput.

Fragile Objects

One key insight derived from our work is the nuanced impact of moving into contact with nonzero velocity, contingent upon factors such as the precision of object modeling and the fragility of manipulated objects. While precise object models may mitigate the benefits of impulsive contact in certain scenarios, the variability and fragility of manipulated objects in real-world industry applications render impact-aware control methods highly advantageous, particularly for poorly understood and non-fragile objects.

However, it is essential to acknowledge the inherent limitations and considerations associated with impact-aware strategies. The maximum allowable impact speed plays a crucial role, as excessively high impact velocities may lead to undesirable consequences, especially for fragile objects. Therefore, a careful balance between the predictability of object behavior and the potential impact forces must be maintained to maximize the effectiveness of impact-aware control methods.

Robustness

In this study, the selection of boxes for investigation aimed to mirror objects commonly encountered in industrial settings. We conducted tests with various masses, sizes, shapes, and inertial distributions, including objects that were not single rigid bodies. However, in our robustness setup, we limited our examination to boxes with a mass no greater than 2 kg. The decision stemmed from the robot's ability to impulsively pick and toss each object 20 times without failure.

The Franka Panda robot's lower torque limits restricted our experiments to objects weighing no more than 2 kg. However, there are no theoretical constraints preventing the application of this technology to any mc_rtc enabled robotic platform with larger torque limits, such as the Kuka IIWA. It's important to note that utilizing this technology on a system with lower torque limits, effectively highlights the advantages of the control structure using mc_rtc. Operating close to the hardware limitations of the robot represents a significant real-world challenge.

It's worth noting that while the dual-arm soft pad design proved effective for a broad range of objects, there are instances where it may not be ideal. For example, bins designed for lifting rather than grasping under pressure, as depicted in Figure 38, present challenges. Their deformation and narrowing towards the top indicate a lack of structural integrity for compression-based grasping.

Similarly, small, irregularly shaped, soft objects wrapped in plastic may be better suited for single gripper or suction cup gripper systems, developed elsewhere in the consortium. Nevertheless, our experimental setup exhibited remarkable robustness, enabling rapid grabbing and tossing of a wide array of objects.

It's essential to highlight that the adoption of this technology does not preclude the use of other end-effector or robotics systems employing alternative grasping approaches for objects unsuitable for compression-based manipulation in industrial settings.

Figure 38: This is an example of a flexible box which is not designed for lifting in compression. (left) Box with out contact. (right) Box after gentile compressive contact. Thus, it would not be well suited for dual arm I.AM. grabbing. Instead it is designed to be held at the handles as a tensile load This type of task requires different end-effectors.

Pre-grasping Manipulation

In this study, our goal was to measure the advantages of utilizing impact, enabled by integrating the scientific advancements achieved within the consortium, for the dual-arm pick-and-toss task. Thus, we specifically targeted scenarios where the box is prepared for grasping. However, in depalletizing processes, preparatory actions, typically undertaken by human workers, are often necessary to position the box for grasping. While aspects of this were explored in the Grab project, they weren't fully integrated into the reported result. Our main emphasis was on quantifying the benefits of integrating the technology developed within the consortium – I.Model, I.Learn, I.Sense, and I.Control, within a unified platform.

Nevertheless, this doesn't rule out the direct extension of the theory and software developed here to the advancement of pre-grasp maneuvers, such as tilting, which can directly benefit from the software developed within this consortium integration.

Summary

In summary, our study underscores the transformative potential of impact-aware technologies in revolutionizing robotic logistics operations. By addressing challenges associated with conventional quasi-static methods and leveraging controlled impacts, these technologies offer significant improvements in task efficiency and energy utilization, thereby paving the way for enhanced productivity and operational effectiveness in warehouse and distribution center environments.

5 CONCLUSION

In conclusion, this report highlights the successful integration of impact-aware technologies and their application in logistics scenarios, with a specific focus on grabbing and depalletizing tasks involving dual-arm robotic systems. Through deliberate utilization of intentional collisions, our consortium has demonstrated the superior speed and energy efficiency achievable with impact-aware robotics, surpassing traditional approaches reliant on quasi-static interactions with objects or environments.

The deliberate generation of desired impacts, previously avoided in classical robotics, poses challenges across modeling, planning, sensing, and control domains. Addressing these challenges, we have integrated the four main components of the I.AM. project: *I.Model*, *I.Learn*, *I.Sense*, and *I.Control*. Through this integration, we have tackled the complexities associated with impact-aware robotics, paving the way for enhanced performance and adaptability in logistics operations.

This report emphasizes the advantages of our proposed approach through extensive experimentation and systematic comparison between classical grabbing techniques and integrated impact-aware strategies. These findings underscore the transformative potential of impact-aware technologies in revolutionizing robotic logistics operations.

A Additional Boxes Results

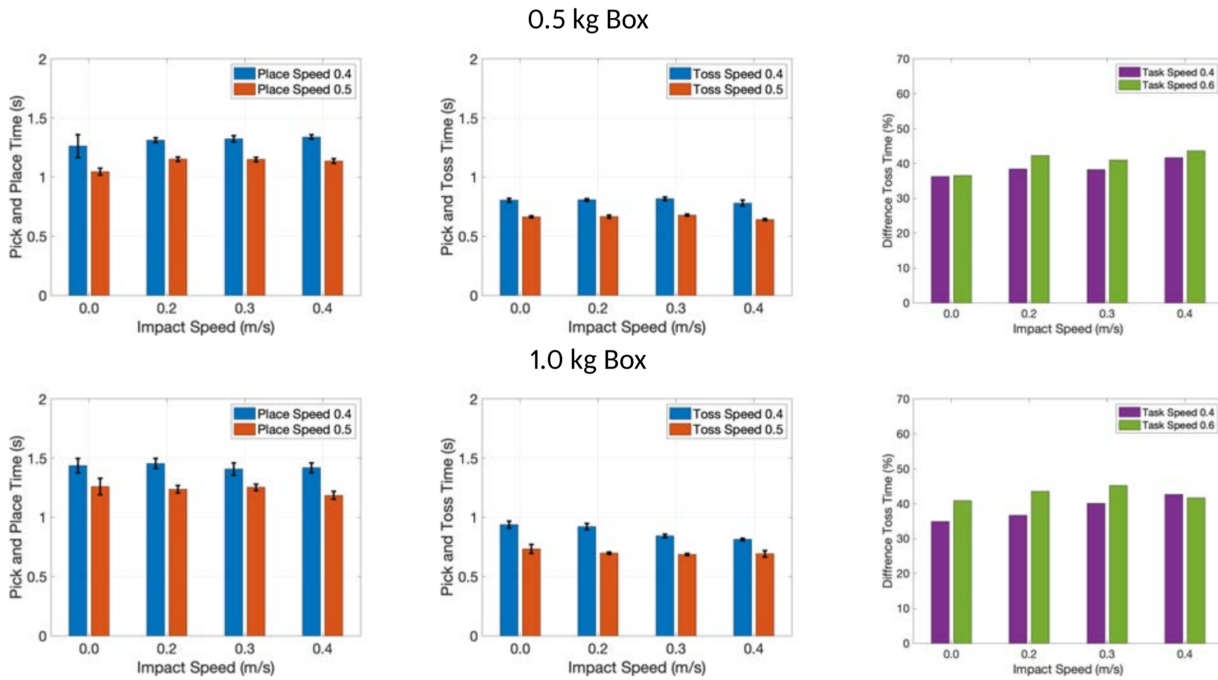


Figure 39: Average Task Time: (top row) 0.5 kg box (bottom row) 1.0 kg box. (left column) Pick and place task times, (middle column) pick and toss task times, (right column) percent difference between the pick and place vs the pick and toss task times. Note that a positive percent difference indicates how much faster the toss task was than the pick task. The bars are grouped by impact speed, within a group of bars. Accordingly, the color denotes the different task speeds according to the legend. The error bars indicated plus or minus one standard deviation from the mean.

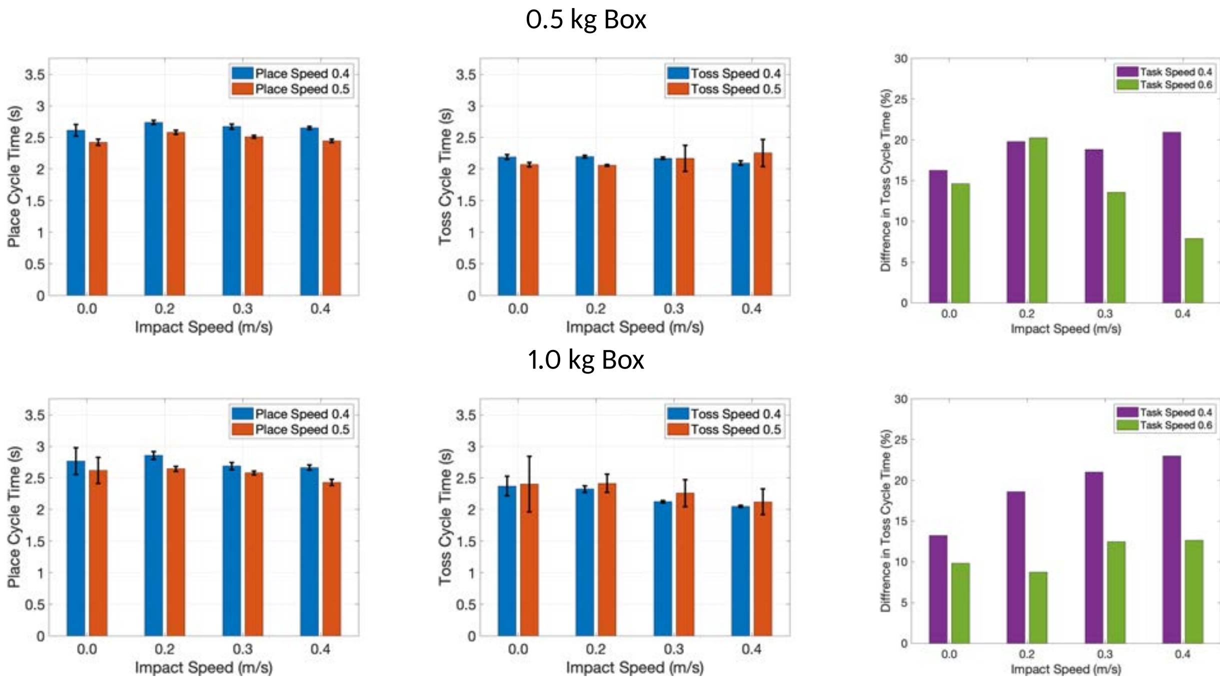


Figure 40: Average Cycle Time: (top row) 0.5 kg box (bottom row) 1.0 kg box. (left column) Place cycle time times, (middle column) toss cycle times, (right column) percent difference between the pick and place vs the pick and toss cycle times. The bar colors and organization are consistent with that of Figure 39.

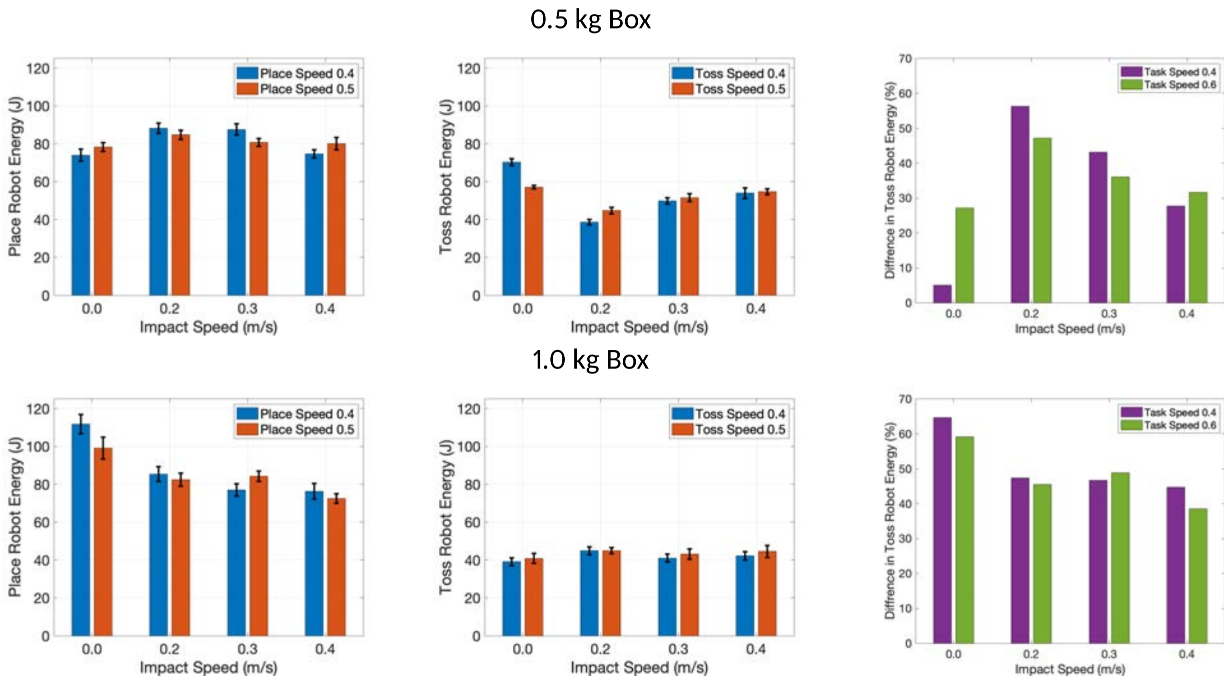


Figure 41: Average Robot Energy: (top row) 0.5 kg box (bottom row) 1.0 kg box. (left column) Place robot energy, (middle column) toss robot energy, (right column) percent difference between the place and the toss energy. The bar colors and organization are consistent with that of Figure 39.

Pre-impact time (s): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	0.207 ± 0.009	0.208 ± 0.012	0.147 ± 0.010	0.113 ± 0.006
Task Speed 0.6	0.220 ± 0.008	0.177 ± 0.011	0.133 ± 0.009	0.108 ± 0.008
Task Speed 0.8	0.220 ± 0.011	0.182 ± 0.011	0.135 ± 0.010	0.114 ± 0.006

Pre-impact time (s): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	0.239 ± 0.011	0.219 ± 0.015	0.163 ± 0.020	0.174 ± 0.040
Task Speed 0.6	0.224 ± 0.009	0.200 ± 0.024	0.172 ± 0.027	0.156 ± 0.014
Task Speed 0.8	0.236 ± 0.012	0.212 ± 0.018	0.158 ± 0.027	0.156 ± 0.019

Pre-impact time (s): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	0.188 ± 0.015	0.162 ± 0.014	0.116 ± 0.008	0.115 ± 0.007
Task Speed 0.6	0.177 ± 0.015	0.159 ± 0.009	0.114 ± 0.006	0.115 ± 0.012
Task Speed 0.8	0.157 ± 0.017	0.136 ± 0.027	0.112 ± 0.018	0.111 ± 0.016

Table 4: Pre-contact time mean and standard deviation for each condition.

Pick and place time (s): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	1.263 ± 0.095	1.312 ± 0.021	1.324 ± 0.026	1.340 ± 0.020
Task Speed 0.5	1.046 ± 0.029	1.153 ± 0.019	1.150 ± 0.018	1.137 ± 0.020

Pick and place time (s): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	1.436 ± 0.060	1.456 ± 0.041	1.408 ± 0.052	1.418 ± 0.041
Task Speed 0.5	1.260 ± 0.070	1.237 ± 0.031	1.253 ± 0.028	1.187 ± 0.032

Pick and place time (s): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	1.443 ± 0.040	1.508 ± 0.018	1.463 ± 0.028	1.425 ± 0.034
Task Speed 0.5	1.257 ± 0.030	1.329 ± 0.092	1.263 ± 0.009	1.181 ± 0.039

Table 5: Pick and place time mean and standard deviation for each condition.

Pick and toss time (s): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	0.806 ± 0.016	0.808 ± 0.011	0.818 ± 0.014	0.781 ± 0.024
Task Speed 0.5	0.664 ± 0.008	0.666 ± 0.013	0.679 ± 0.009	0.641 ± 0.009

Pick and toss time (s): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	0.936 ± 0.024	0.922 ± 0.025	0.843 ± 0.014	0.814 ± 0.011
Task Speed 0.5	0.733 ± 0.038	0.698 ± 0.010	0.687 ± 0.010	0.692 ± 0.028

Pick and toss time (s): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	0.888 ± 0.012	0.988 ± 0.017	0.995 ± 0.024	0.972 ± 0.023
Task Speed 0.5	0.722 ± 0.009	0.813 ± 0.027	0.808 ± 0.021	0.777 ± 0.018

Table 6: Pick and toss time mean and standard deviation for each condition.

Place cycle time (s): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	2.613 ± 0.094	2.738 ± 0.030	2.672 ± 0.040	2.650 ± 0.023
Task Speed 0.5	2.423 ± 0.050	2.580 ± 0.034	2.509 ± 0.021	2.446 ± 0.028

Place cycle time (s): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	2.767 ± 0.213	2.855 ± 0.062	2.687 ± 0.055	2.664 ± 0.041
Task Speed 0.5	2.619 ± 0.206	2.644 ± 0.040	2.580 ± 0.030	2.427 ± 0.048

Place cycle time (s): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	2.601 ± 0.045	2.822 ± 0.027	2.729 ± 0.034	2.645 ± 0.050
Task Speed 0.5	2.427 ± 0.028	2.579 ± 0.053	2.541 ± 0.016	2.376 ± 0.058

Table 7: Place cycle time mean and standard deviation for each condition.

Toss cycle time (s): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	2.188 ± 0.039	2.196 ± 0.023	2.169 ± 0.019	2.094 ± 0.033
Task Speed 0.5	2.069 ± 0.035	2.057 ± 0.015	2.169 ± 0.205	2.252 ± 0.213

Toss cycle time (s): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	2.401 ± 0.083	2.323 ± 0.049	2.122 ± 0.019	2.051 ± 0.016
Task Speed 0.5	2.402 ± 0.440	2.414 ± 0.143	2.258 ± 0.215	2.120 ± 0.203

Toss cycle time (s): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	2.051 ± 0.019	2.308 ± 0.025	2.267 ± 0.038	2.243 ± 0.035
Task Speed 0.5	2.255 ± 0.117	2.085 ± 0.114	2.080 ± 0.030	2.039 ± 0.089

Table 8: Toss cycle time mean and standard deviation for each condition.

Place robot energy (J): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	74.0 ± 3.2	88.2 ± 2.7	87.5 ± 3.0	74.6 ± 2.2
Task Speed 0.5	78.3 ± 2.4	84.7 ± 2.4	80.7 ± 2.1	80.0 ± 3.3

Place robot energy (J): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	111.7 ± 5.0	85.4 ± 3.8	77.0 ± 3.2	76.2 ± 4.2
Task Speed 0.5	99.0 ± 5.8	82.4 ± 3.4	84.2 ± 2.7	72.5 ± 2.6

Place robot energy (J): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	78.3 ± 2.6	85.7 ± 1.7	80.1 ± 2.1	75.0 ± 3.2
Task Speed 0.5	81.9 ± 1.7	84.2 ± 2.9	100.8 ± 1.6	72.7 ± 2.7

Table 9: Place robot energy mean and standard deviation for each condition.

Toss robot energy (J): Box 0.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	70.2 ± 1.8	38.5 ± 1.5	49.8 ± 1.6	53.9 ± 2.8
Task Speed 0.5	57.0 ± 0.9	44.7 ± 1.7	51.5 ± 2.1	54.7 ± 1.6

Toss robot energy (J): Box 1.0 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	39.1 ± 2.1	44.9 ± 2.1	41.0 ± 2.1	42.2 ± 2.3
Task Speed 0.5	40.9 ± 2.6	44.9 ± 1.7	43.1 ± 2.7	44.5 ± 3.1

Toss robot energy (J): Box 1.5 kg

	Impact Velocity (m/s)			
	0.0	0.2	0.3	0.4
Task Speed 0.4	45.8 ± 1.5	47.2 ± 1.2	45.5 ± 0.9	49.7 ± 3.7
Task Speed 0.5	48.2 ± 0.9	43.3 ± 2.2	46.1 ± 1.5	46.4 ± 1.5

Table 10: Toss robot energy mean and standard deviation for each condition.

B REFERENCES

[1] Nils Boysen, René de Koster, and David Füßler. The forgotten sons: Warehousing systems for brick-and-mortar retail chains. *European Journal of Operational Research*, 288(2):361–381, 2021.

[2] Mohd Javaid, Abid Haleem, Ravi Pratap Singh, Shanay Rab, and Rajiv Suman. Significant applications of cobots in the field of manufacturing. *Cognitive Robotics*, 2:222–233, 2022.

[3] Harshit Khurana, Michael Bombile, and Aude Billard. Learning to hit: A statistical dynamical system based approach. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9415–9421, 2021.

[4] Harshit Khurana and Aude Billard. Motion planning and inertia-based control for impact aware manipulation. *IEEE Transactions on Robotics*, pages 1–16, 2023.

[5] Matthew T Mason. Toward robotic manipulation. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:1–28, 2018.

[6] Xianyi Cheng, Yifan Hou, and Matthew T Mason. Manipulation with suction cups using external contacts. In *International Symposium of Robotics Research*, pages 692–708. Springer, 2022.

[7] A Oliva, F Nordfeldth, M Jongeneel, R Duarte, and S Eisinger. Scenario 2 (BOX) Report. Technical Report D5.4, H2020 EU Project I.AM. Consortium, 2024.

[8] A Oliva, M J Jongeneel, and A Saccon. A compact 6D suction cup model for robotic manipulation via symmetry reduction. *Submitted to: IEEE Transactions on Robotics (T-RO)*, March 2023.

[9] Yuki Shirai, Devesh K Jha, and Arvind U Raghunathan. Robust pivoting manipulation using contact implicit bilevel optimization. *arXiv preprint arXiv:2303.08965*, 2023.

[10] Mathew Halm and Michael Posa. A quasi-static model and simulation approach for pushing, grasping, and jamming. In *Workshop on the Algorithmic Foundations of Robotics*, pages 491–507, 2020.

[11] Yuki Shirai, Devesh K. Jha, Arvind U. Raghunathan, and Dennis Hong. Tactile tool manipulation. In *IEEE International Conference on Robotics and Automation*, 2023.

[12] Michael Bombile and Aude Billard. Dual-arm control for coordinated fast grabbing and tossing of an object: Proposing a new approach. *IEEE Robotics & Automation Magazine*, 29(3):127–138, 2022.

[13] Alessandro Saccon, Nathan van de Wouw, and Henk Nijmeijer. Sensitivity analysis of hybrid systems with state jumps with application to trajectory tracking. In *Conference on Decision and Control*, pages 3065–3070. IEEE, 2014.

[14] Jari J. van Steen, Nathan van de Wouw, and Alessandro Saccon. Robot control for simultaneous impact tasks through time-invariant reference spreading. In *2023 American Control Conference (ACC)*, pages 46–53, 2023.

[15] Jari J. van Steen, Abdullah Coşgun, Nathan van de Wouw, and Alessandro Saccon. Dual arm impact-aware grasping through time-invariant reference spreading control. *IFAC-PapersOnLine*, 56(2):1009–1016, 2023. 22nd IFAC World Congress.

[16] Jari van Steen, Gijs van den Brandt, Nathan van de Wouw, Jens Kober, and Alessandro Saccon. Quadratic programming-based reference spreading control for dual-arm robotic manipulation with planned simultaneous impacts. *arXiv preprint arXiv:2305.08643*, 2023.

[17] Yuquan Wang and Abderrahmane Kheddar. Impact-friendly robust control design with task-space quadratic optimization. In *Proceedings of Robotics: Science and Systems*, volume 15, page 32, Freiburg, Germany, 24-26 June 2019.

[18] Yuquan Wang, Niels Dehio, and Abderrahmane Kheddar. On inverse inertia matrix and contact-force model for robotic manipulators at normal impacts. *IEEE Robotics and Automation Letters*, 7(2):3648–3655, 2022.

[19] Yuquan Wang, Niels Dehio, and Abderrahmane Kheddar. Predicting post-impact joint velocity jumps on kinematics controlled manipulators. *IEEE Robotics and Automation Letters*, 7(3):6226 – 6233, 2022.

[20] Yuquan Wang, Niels Dehio, Arnaud Tanguy, and Abderrahmane Kheddar. Impact-aware task-space quadratic-programming control. *International Journal of Robotics Research*, 2023.

[21] Sami Haddadin, Alessandro De Luca, and Alin Albu-Schäffer. Robot collisions: A survey on detection, isolation, and identification. *IEEE Transactions on Robotics*, 33(6):1292–1312, 2017.

[22] Cristovao D Sousa and Rui Cortesao. Physical feasibility of robot base inertial parameter identification: A linear matrix inequality approach. *The International Journal of Robotics Research*, 33(6):931–944, 2014.

[23] Mario Tröbinger, Abdeldjallil Naceri, Xiao Chen, Hamid Sadeghian, and Sami Haddadin. Identification of a generalized base inertial parameter set of robotic manipulators considering mounting configurations. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11502–11508. IEEE, 2023.

[24] Alessandro De Luca and Raffaella Mattone. Actuator failure detection and isolation using generalized momenta. In *2003 IEEE international conference on robotics and automation (cat. No. 03CH37422)*, volume 1, pages 634–639. IEEE, 2003.

[25] Jonathan Vorndamme, Alessandro Melone, Robin Kirschner, Luis Figueredo, and Sami Haddadin. *Safe Robot Reflexes: A Taxonomy-based Decision and Modulation Framework*, [Manuscript submitted for publication, 2024].

[26] F. Voigt, A. Naceri, and S. Haddadin. Manipulator agnostic tactile manipulation and execution monitoring of delicate contact-rich manipulation on task level. [Manuscript in preparation], 2024.

[27] Michael Bombile and Aude Billard. Bimanual dynamic grabbing and tossing of objects onto a moving target. *Robotics and Autonomous Systems*, 167:104481, 2023.