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

I.Learn Report

Dissemination level	Public (PU)
Work package	WP2:Learning
Deliverable number	D2.2
Version	F-1.0
Submission date	02/10/2023
Due date	30/09/2023

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**
Aude BILLARD	EPFL	aude.billard@epfl.ch
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
Ahmed ZERMANE	CNRS	ahmed.zermane@lirmm.fr

Control sheet

Version history			
Version	**Date**	**Modified by**	**Summary of changes**
0.1	15/05/2023	Jos DEN OUDEN	TOC & first contents
0.11	18/06/2023	Michael BOMBILE	General structure and first content
0.12	01/09/2023	Michael BOMBILE	Updated Sections 2.2.3, 2.3.2, 2.3.3
0.13	02/09/2023	Harshit KHURANA	Updated Sections 2.2.1, 2.2.2, 2.3.1
0.14	05/09/2023	Michael BOMBILE	Finished Sections 2.2.3, 2.3.2, 2.3.3
0.14	05/09/2023	Harshit KHURANA	Finished Sections 2.2.1, 2.2.2, 2.3.1
0.14	05/09/2023	Elise JEANDUPEUX	Updated Section 2.5
0.15	06/09/2023	Michael BOMBILE	Updated and Finished Section 3
0.15	06/09/2023	Aude BILLARD	Corrections and comments
0.2	14/09/2023	James HERMUS	Updated Section 2.4
0.3	23/09/2023	Ahmed ZERMANE	Updated Section 2.2.4
0.8	25/09/2023	Michael BOMBILE	Peer-review comments addressed
0.9	28/09/2023	Michael BOMBILE	Revised version ready for submission, quality check
1.0	02/10/2023	Jos DEN OUDEN, Alessandro SACCON	Revised version ready for submission, quality check

Peer reviewers		
	Reviewer name	**Date**
Reviewer 1	James HERMUS	06/09/2023
Reviewer 2	Aude BILLARD	06/09/2023
Reviewer 3	Maarten JONGENEEL	22/09/2023

TABLE OF CONTENTS

ABBREVIATIONS

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

EXECUTIVE SUMMARY

This Deliverable D2.2 (I.Learn report) presents an overview of the publications associated with the I.Learn framework. It focuses particularly on tasks T2.2 and T2.3 and provides summaries of each paper's contributions to each considered task. It highlights the main achievements and discusses the limitations and future directions.

Apart from the executive summary and the conclusion, the report has two main parts built around tasks T2.2 and T2.3 dealing respectively with the determination of postures that prepare robots to generate impacts or release events, and learning impedance and dynamical systems to control impact tasks. Thus, in the first part, the report presents the summary of three publications associated in particular with the generation of the desired impact states, the postures necessary for a desired momentum exchange in hitting tasks, and then the generation of tossing postural states with a dual-arm system. In the second part, the report summarizes publications (four) associated with the design of dynamical systems used for the robust generation of impact, grabbing, and tossing motion.

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 to exploit intentional impacts for manipulation. I.AM focuses on impact aware manipulation in logistics, a new area of application for robotics which will grow exponentially in the coming years, due to socio-economical drivers such as booming of e-commerce and scarcity of labour. 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 the 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, a bin-to-bin application object fast boxing, and a case depalletizing scenario demonstrating object grabbing.

1.2 I.Learn background

I.Learn is the I.AM framework aimed at generating for robots feasible dynamic motions with velocity jumps. It offers a learning-based technology for optimizing pre-impact velocity and posture to achieve post-impact velocities aligned with desired dynamic manipulation objectives, such as hitting, tossing, boxing, or grabbing. This framework is intended to provide reference motions, optimal impedance, task priorities, and contingency plans for fault recovery strategies.

The I.Learn models will be validated on the basis of their ability to generate feasible and computationally faster impact postural states than standard approaches. Moreover, their robustness and real-time ability to adapt to several disturbances of the impact-aware manipulation tasks will be verified in realistic experiments.

1.3 Purpose of the deliverable

The objective of this deliverable D2.2 (I.Learn report) is to present a comprehensive overview of research papers associated with the I.Learn framework. This document offers a concise summary of the main results and algorithmic approaches outlined in these papers. It further highlights the primary accomplishments, limitations, and prospective directions related to I.Learn (T2.2 and T2.3).

1.4 Intended audience

The dissemination level of D2.2 is “public” (PU) – meant for members of the Consortium (including Commission Services) and the general public. This document is designed to also serve as an internal communication for the entire I.AM consortium by providing information on the developments and publications associated with I.Learn.

2 SUMMARY AND RESULTS OF I.LEARN

2.1 I.AM WP2 objectives

We recall that I.AM. project objectives related to WP2 (Learning) are:

OBJ2.1 Learning models of impacts and of object's dynamics resulting from impact, based on high-resolution simulation from Model;

OBJ2.2 Compute postures that prepare the robot to generate impacts and release events with its surrounding with an outcome that is aligned with the user-specified dynamic manipulation goals;

OBJ2.3 Learning robot controller for different types of impact: swiping, tossing, grabbing, boxing;

OBJ2.4 Learning of QP-parameters for multi-contact planning under stability constraints

This deliverable D2.2 mainly focuses on the second and third objectives (OBJ2.2 and OBJ2.3) which concern the generation of impact postural states and the design of impact task controllers, respectively.

2.2 Task 2.2 Impact Posture Generator for Dynamic Manipulation: Overview of publications

The aim of this task is to devise an impact posture generator for dynamic manipulation. It allows to determine what pre-impact posture and speed a robot has to take in order to achieve a desired post-impact speed in the task space. In other words, given the initial state (pose and velocity) of a robotic system, the goal is to find a state trajectory along which the robot will be steered to a given desired impact location and velocity.

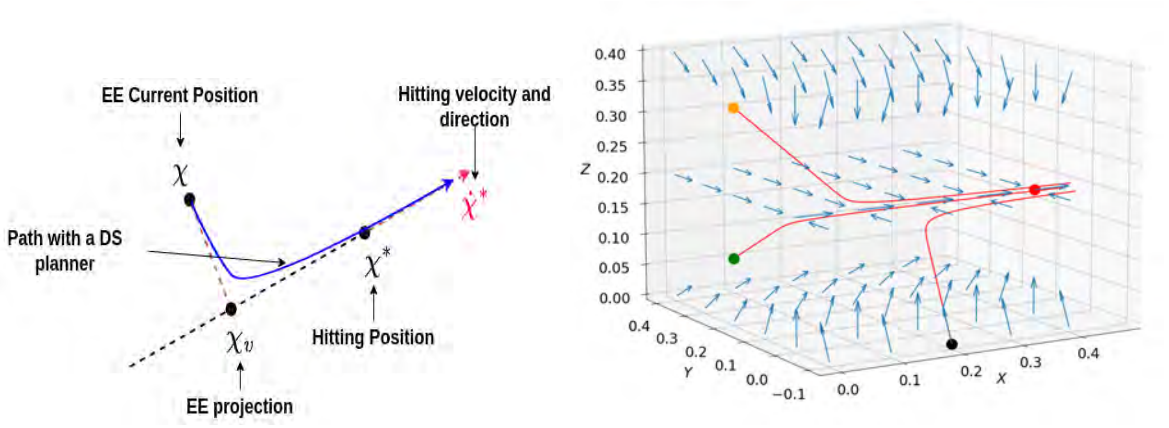
The impact task, unlike the positioning task, is characterized by a transitory state along a trajectory and cannot be solved with a static posture generator. Its problem formulation using standard optimization tools would lead to a semi-infinite programming (SIP) problem that is known to be hard to solve efficiently and certainly not in real-time. To address this problem, two possible approaches have been considered in T2.2, as reported in Deliverable D2.1. The first is a data-driven approach that generates impact postural states from simulated experiments of the intended use-cases. The second approach combines planning and optimization and decouples each impact problem into two main phases: (i) planning the desired impact state from the intended task (use-case) and (ii) planning the reaching of the desired impact state.

Although some technical details related to T2.2 were previously reported in Deliverable 2.1 and Deliverable 5.3, this section summarizes the most important results of publications associated with T2.2.

2.2.1 Publication: "Learning to Hit: A statistical Dynamical System based approach"

2.2.1.1 Approach Summary

This paper [1] proposes a manipulation scheme based on learning the motion of objects after being hit by a robotic end-effector. The motion of the object after being subjected to a hitting force or an impulse is hard to predict due to numerous reasons. The reasons can be modelling difficulties such as friction which depends on microscopic interations of the object and the sliding surface, coefficient of restitution



(a) (b)

Figure 1: (a) Design intuition for the dynamical system (DS) as a motion generator. The end-effector (EE) moves toward its projection on the desired direction and a constant flow is added. The effect of the two different flows is controlled by a weighting parameter, which depends on the distance of the end effector from its projection. (b) The vector field showing the flow of the end effector as in Eq. 1. The red marker shows the position of the object that needs to be hit. The figure also shows three different paths the robot takes from three different initial positions (in orange, green and black)

and, the contact between the robot and the object. Furthermore the control system has its uncertainties in following the desired trajectory to hit the box and this can lead to errors in where the box is being hit. This leads to simple Newtonian model to be prone to errors in predicting the motion of the object after being subject to an impulse. An estimate of the object dynamics under friction and collisions is learnt and used to predict the desired hitting parameters (speed and direction), given the initial and desired location of the object. The learning method is presented in 2.3.1. Based on the obtained hitting parameters, the desired pre-impact velocity of the end-effector is generated using a stable dynamical system of the form

$$\dot{\chi} = f(\chi) = \alpha(\chi)\dot{\chi}^* + (1 - \alpha(\chi))[A(\chi - \chi_v)] \tag{1}$$

where χ_v is projection of the end effector's position on the hitting line, $\alpha(\chi)$ is the weighting factor between the two components of the dynamical system: first that drives the robot at the desired speed and second, that drives the robot towards the hitting line.
This allows for the object to be positioned at a desired location outside the physical workspace of the robot.

2.2.1.2 Limitations and Future perspectives

From the motion planning perspective, we generate a motion using a dynamical system. This motion currently requires us prior understanding of what end-effector velocities are possible. Since the achievable velocities of the robot are dependent on its configuration, achieving a desired speed and a desired inertia needs to be designed carefully. We are working on achieving this through an optimisation problem which will be implemented in the next work.

2.2.2 Publication: "Motion Planning and Inertia Based Control for Impact Aware Manipulation"

2.2.2.1 Approch Summary

In this paper, we propose a metric called hitting flux which is used in the motion generation and controls for a robot manipulator to interact with the environment through a hitting or a striking motion. The process of collision is an exchange of momentum between the colliding objects and hence the post-impact state of the object being hit depends on the inertia of the robot, speed of the robot and its own inertia. Given the task of placing a known object outside of the workspace of the robot, the robot needs to come in contact with it at a non-zero relative speed. The configuration of the robot and the speed at contact matter because they affect the motion of the object. The physical quantity called hitting flux depends on the robot's configuration, the robot speed and the properties of the environment. We combine maximising manipulability metric with control of directional inertia to achieve the desired hitting flux. Maximising manipulability allows the robot to achieve high end effector velocities. This is ensured by prioritizing the configurations that have high velocity manipulability metric, along the hitting direction, $\hat{h} \in \mathbb{R}^3$. For achieving desired directional inertia, the joint configuration is allowed to move in the Null Space of the motion task. This changes the joint velocities towards the configuration with the desired directional inertia if the redundancy allows for it.
Thus, the described control system leading to directional inertia values close enough to the desired values, with the speed of the robot being controlled by a torque-based velocity tracking controller, allows us to achieve the desired hitting flux. The bloc diagram for the hitting system is depicted in Figure 2.

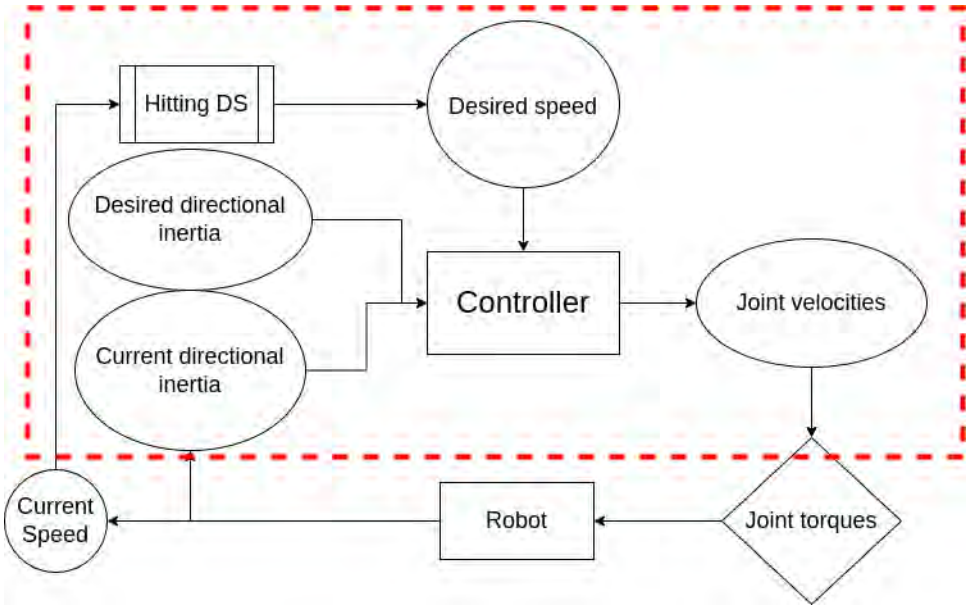


Figure 2: The figure depicts the block diagram for the hitting system.

2.2.2.2 Limitations and Future perspectives

From the control perspective, it requires us prior understanding of what directional inertia values are possible due to the initial configuration of the robot. Currently, the formulation of the hitting flux in terms of directional inertia has a few assumptions on the numerical values in the inertia matrix. This

can be further generalized by using effective inertia in the hitting flux, which can further utilize the Null Space in the joint space of the robot. This will also be implemented in our upcoming work. Secondly, we assume the coefficient of restitution as a given parameter, and to overcome this we are looking into different learning procedures.

2.2.3 Publication: "Bimanual dynamic grabbing and tossing of objects onto a moving target"

2.2.3.1 Approach Summary

As part of this publication [3], a framework that focuses on the computation of robot posture for dynamic release tasks with a dual-arm robot was proposed. More precisely, this posture generator algorithm enables a dual-arm robotic system to toss precisely a grabbed object onto a moving target. The proposed framework addresses the problem of explicit computation of the tossing parameters of the object and more importantly their corresponding robot postural states.

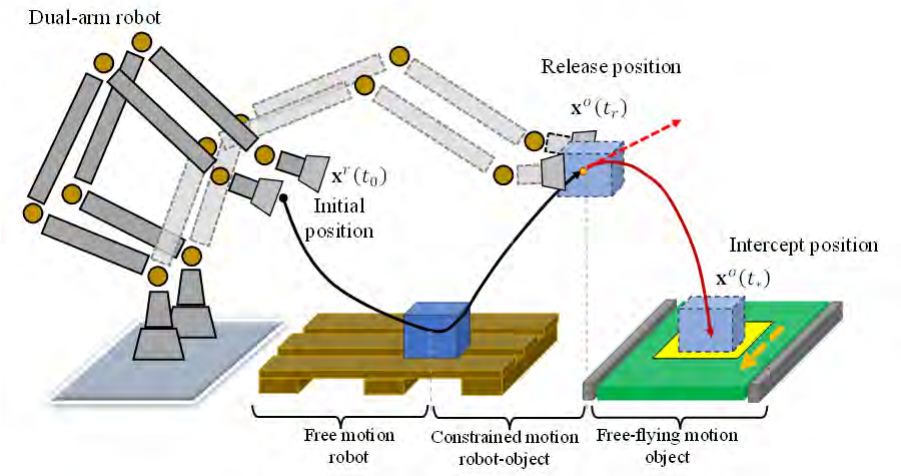


Figure 3: Illustration of dual-arm robotic depalletizing task with fast grabbing and tossing of an object (blue) onto a moving target (yellow). The overall motion can be split into three main phases; the free robot motion, the constrained robot–object motion, and the object free-flying motion before interception of the target at landing. The initial robot's posture is shown in dark grey whereas a release posture achieving the task is illustrated in light grey

An illustration of the posture generation task of a dual-arm robot in a depalletizing task is given in Figure 3. Given the desired landing position for the object, the proposed algorithm estimates the release parameters necessary to throw the object at the desired landing location. Estimating the release parameters is nothing but solving the inverse throwing problem that consists of determining a feasible release state leading to the desired landing state. To that end, we developed a mixed learning-optimization method that computes feasible release positions and velocities of the object and their associated postural states of the robot. To address the redundancy problem inherent to the throwing task, we adopted a strategy that seeks solutions yielding minimum throwing efforts. Hence, using Gaussian mixture regression (GMR) and data from the object's free-flying dynamics with nonlinear drag, we learned an inverse throwing map that generates minimum release velocities of the object for given relative release positions (relative to the landing position). Then, this learned map is embedded into a kinematics-based bi-level optimization that computes the associated postural release states of the dual-arm robot while

enforcing the feasibility constraints both in position and velocity and also the dual-arm coordination constraints necessary to maintain the grasp of the object. The proposed approach is summarized in Figure 4.

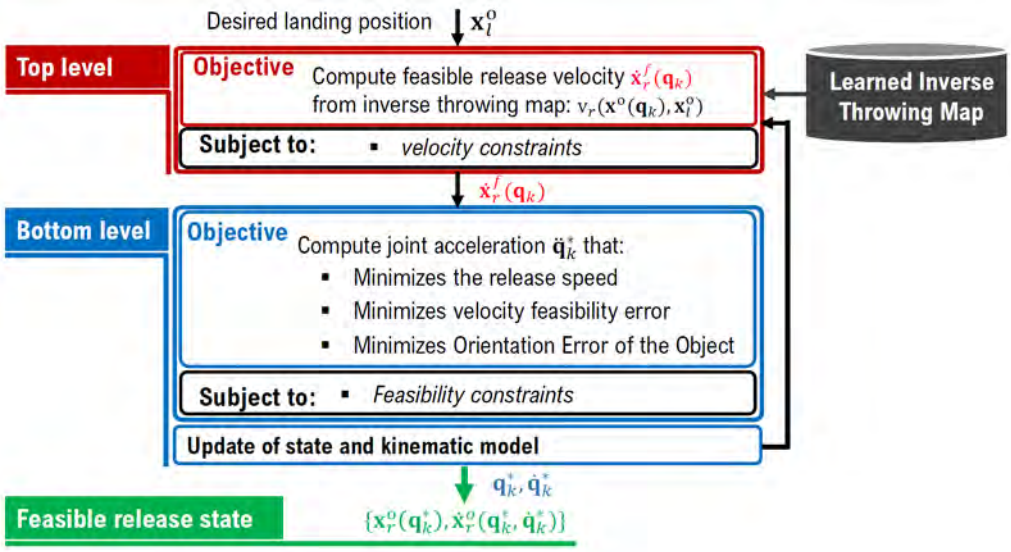


Figure 4: Illustration of the mixed learning-optimization method that computes feasible release positions and velocities of the object and their associated postural states of the robot. The bi-level optimization runs iteratively and computes at the top level the robot's feasible release velocity closest to the minimum velocity predicted by the inverse throwing map at the current configuration. At the bottom level, the algorithm computes joint accelerations leading to configurations minimizing the difference between the previously obtained feasible release velocity and the prediction of the inverse throwing map. The algorithm also enforces not only the joint position and velocity limits, but also the dual-arm coordination constraints necessary to maintain the grasp of the object.

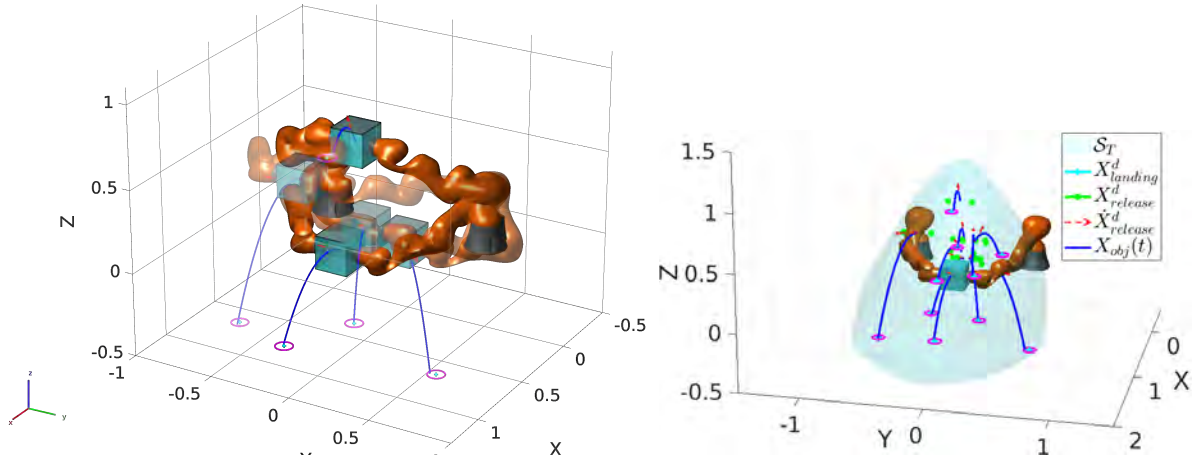
2.2.3.2 Main achievements, Limitations, and Future perspectives

The posture generator presented in the publication [3], although developed for tossing tasks with a dual-arm robotic system, is based on a conceptual framework that can also be applied to other impact or dynamic release tasks, such as tossing with multi (more than 2) arm system. The learning component of the proposed approach encodes a closed-form solution of a sub-problem and thereby allows fast computation. Prior to its real-world validation, the posture generator was extensively evaluated in simulation. It was assessed on a set of 10^5 uniformly distributed 3D landing positions within and outside of the dual-arm workspace (within a radius of $0.2 - 1.75\,m$ and within a cone of $\pm\frac{\pi}{3} rad$ around the forward horizontal axis).

The posture generator algorithm could determine the throwing reachability of each generated point by checking whether or not it could find a feasible release state. The results of such an assessment allowed us to derive a closed-form model of the dual-arm robot's tossable space (throwing workspace) for a given object by modeling the distribution of all reachable points. Finally, we validated experimentally the proposed dual-arm generator of tossing postural states on two 7-DoF KUKA robotic arms with

dynamical system-based motion generation described in subsection 2.3.2. The accuracy and the real-time ability to generate tossing posture were demonstrated. Figure 5 illustrates a sample of five 3D and joint-space release configurations and the associated free-flying trajectories of the object to the desired landing position. The corresponding values of the release postural states (joint-space release positions and velocities) plus five more are shown in Figure 6.

The proposed posture generator, however, only partially addresses the objectives of task T2.2. It allows the determination of a feasible postural state necessary to perform the impact or release task, but it does generate the state trajectory along which the robot should evolve to reach the desired postural state. This is because the proposed posture generator was developed under the assumption of kinematic feasibility, valid state trajectories would require enforcing the dynamic feasibility constraints.



(a) (b)

Figure 5: Illustration of generated 3D and joint-space release configurations of the object and the dual-arm robot with the associated free-flying trajectories of the object to the desired landing position. (a)- five individual feasible release configurations (b)- feasible release 3D positions (green dot) and corresponding object trajectories (blue) to the desired landing position (cyan points).

2.2.4 Publication: Planning Impact-Driven Logistic Processes

2.2.4.1 Approach Summary

This publication introduces an innovative approach designed to plan and manage robotic processes that involve impact, emphasizing their relevance in logistics. The method proposed in this study comprises a two-tiered model-based strategy, as illustrated in Figure 7. The initial component of this strategy is centered on generating trajectories for a robotic arm to transition seamlessly from one state to another. It primarily utilizes a kinodynamic Bi-RRT sampling-based planner while adhering to joint-level constraints. The second component, tailored to the specific requirements of each process, establishes a connection between the desired impact goal within the task space and an intermediate point in the joint space. These tasks are formulated with certain assumptions, such as knowing the inertial characteristics of the objects being handled. The outcomes of this study underscore the system's capacity to execute these tasks with precision, with a primary focus on managing impacts and manipulating objects.

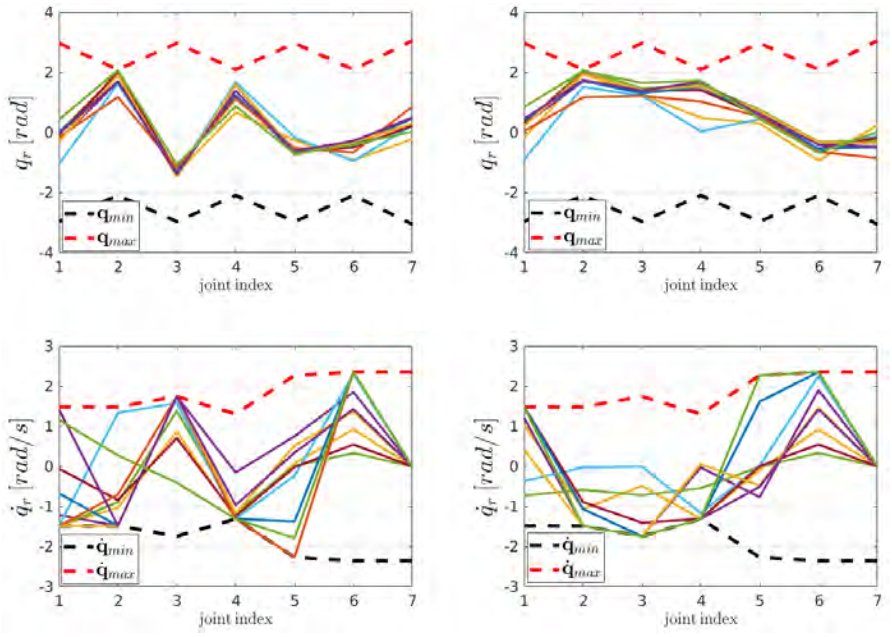


Figure 6: Release postural states (joint-space release positions and velocities) corresponding to the ten examples of Figure 5b. top: release joint positions for left and right robotic arms; bottom: release joint velocities for left and right arms. The joint limits for position and velocity are shown in red and black dashed lines, respectively.

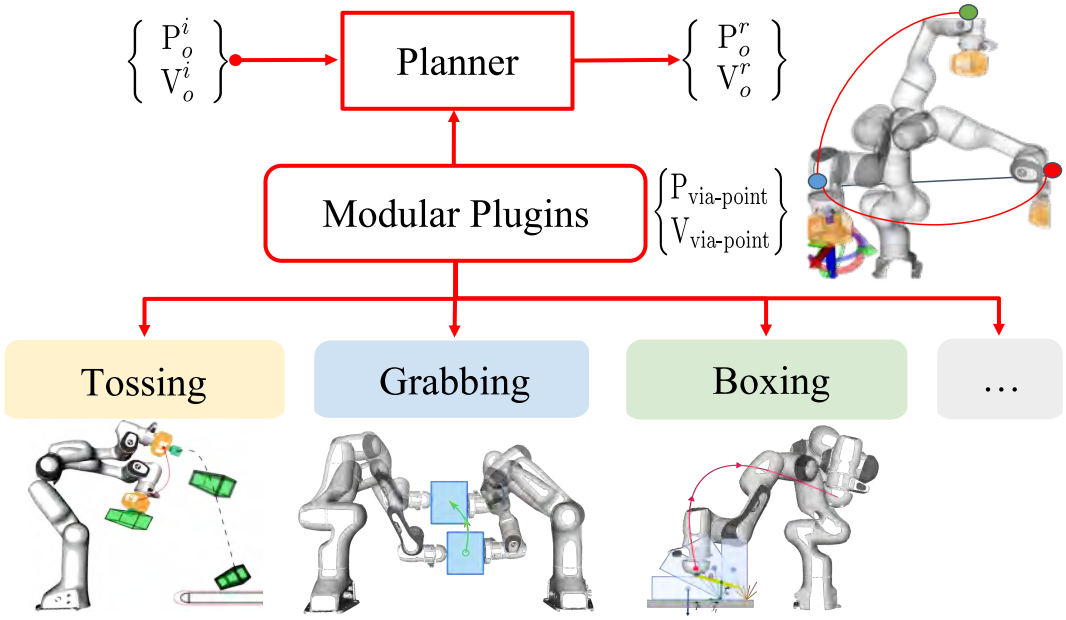


Figure 7: Proposed motion generator for scenarios with impact.

2.2.4.2 Main achievements, Limitations, and Future perspectives

The main achievements of this work include successful Real-world experiments with Panda robots that demonstrate precise performance in tossing, grabbing, and boxing tasks within logistics contexts.

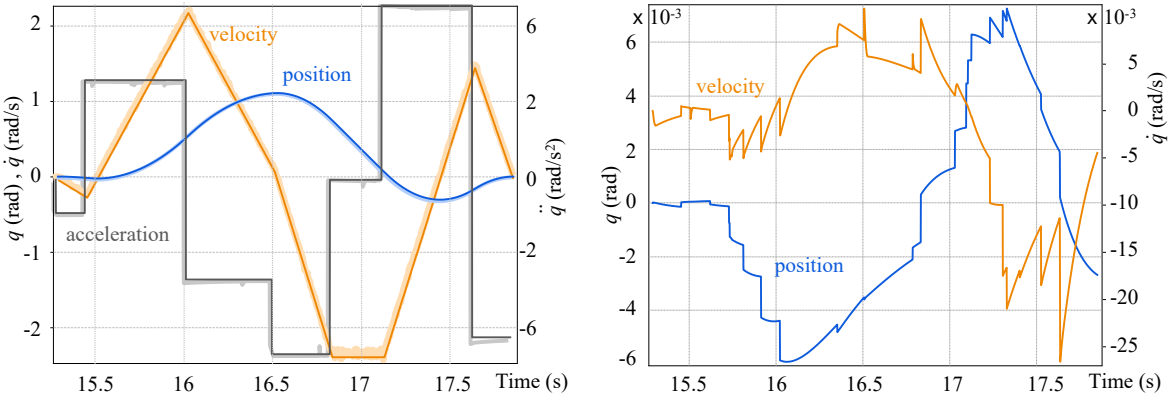


Figure 8: Position, velocity, and acceleration tracking for joint 1: desired (dark) and real (shaded), (left). Position and velocity tracking errors (right).

In Figure 8, we show the tracking performance of the QP controller over a trajectory sample. These results are similar for tasks.

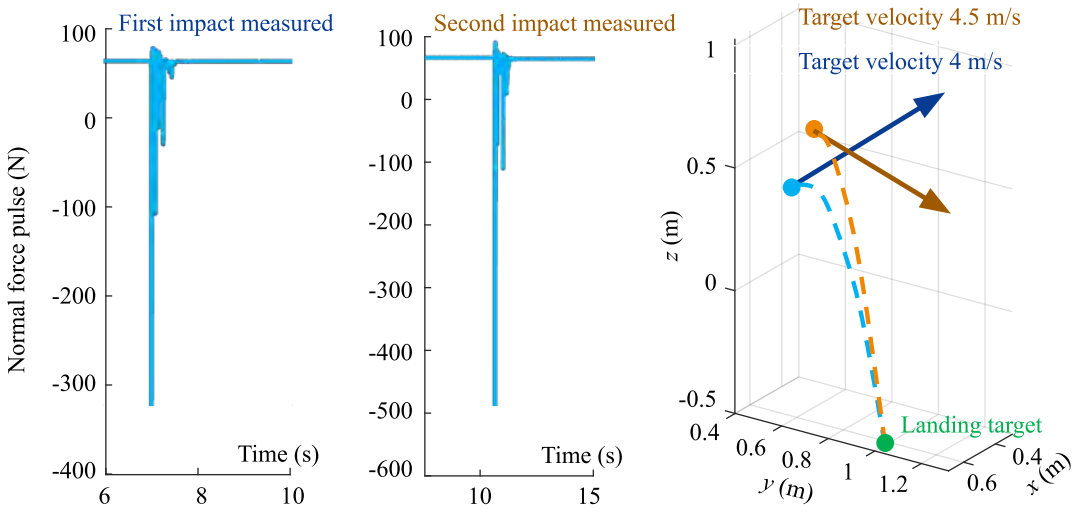


Figure 9: Same landing target with different impact velocities.

In Figure, 9, a throwing task with different objects impacting velocities is used. For the boxing case (see Figures 10, 11), force measurements at the moment of impact and the corresponding task velocity tracking with velocity jumps during impact are shown. Also, one point to mention is that we can achieve all the tasks combined (see Figure 12) in one process that could be designed according to the need. However, the work has several limitations. The experiments are conducted in well-calibrated settings without using vision for object tracking, which limits the approach's robustness in uncontrolled environments. The knowledge of object shape and inertia parameters is assumed, which may only sometimes

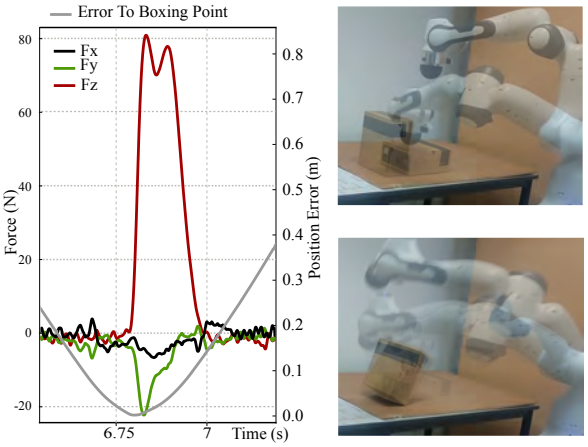


Figure 10: Boxing sequence (right) and measured impact forces (consequent to desired impacting velocity) with position error to impacting point M (left).

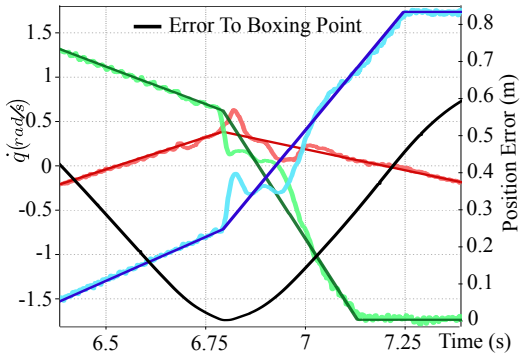


Figure 11: Desired joint velocity for three joints, $i = \{2, 3, 4\}$, dark colors (red, blue, green), and the real measured corresponding velocities by the encoders with light colors.

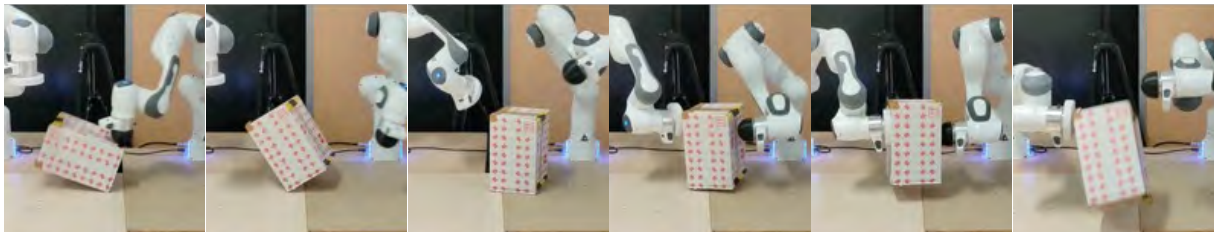
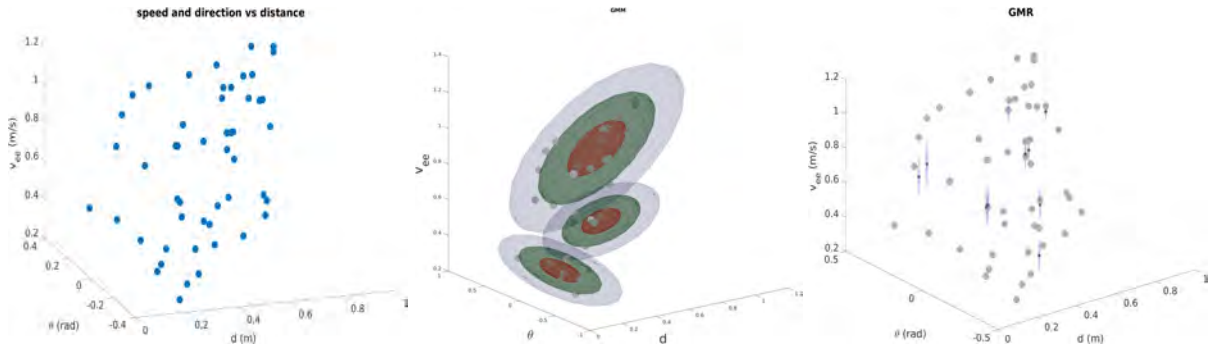


Figure 12: Sequencing boxing, grabbing, and tossing in one run.

be available in logistics scenarios. Future work should consider online inertia parameter estimation and integration with vision systems for improved performance and robustness.
Future perspectives for this research include extending the approach to online optimization planning and enhancing robustness to uncertainties. Additional processes like impact assemblies and real-time object catching should also be considered. The work lays the foundation for more advanced and adaptive robotic processes in logistics and beyond, promising further advancements in automation and industrial applications.

2.3 T2.3 Learning of Impedance and Dynamical Systems for Control with Impacts: Overview of publications

The I.AM task T2.3 requires the learning of robot controllers for different types of impacts such as hitting, tossing, boxing, and grabbing. It also requires learning appropriate impedances and their modulation strategies to perform the desired impact or release tasks while mitigating the impact-induced state jumps.
This section summarizes the main concepts of the publications associated with dynamical systems-based controllers for impact-aware hitting, grabbing, and tossing. Some technical details of the tossing task controller were provided in Deliverable 5.3.



(a) (b) (c)

Figure 13: (a) Collected data: for different end effector speeds and direction of hit, we measure the displacement of the object. (b) GMM Model: Here we see the data modeled using two gaussians in a GMM. Each gaussian is shown with three iso-contour ellipsoids. (c) An example of speed prediction for the cross validation test data - ellipsoid in blue depicts one standard deviation in the prediction value.

2.3.1 Publication: "Learning to Hit: A statistical Dynamical System based approach"

2.3.1.1 Summary of main achievements

In the paper [1], we have developed a preliminary learning methodology to predict how an object moves upon impact from a robot. The robot hits the object at nearly the same configuration each time and hence the object motion can be controlled by controlling the speed and the direction in which the robot hits the object. The motion of the robot as described in the SubSec. 2.2 is implemented on a KUKA LBR iiwa 7 robot. It impacts an object of known mass and the data for hitting speed, hitting direction and the distance covered by the object is recorded. This data is then used to model a generative learning model using Gaussian Mixture Models. This is total probability of the data, i.e. $P(d, \theta, v_{ee})$ A Gaussian Mixture Model is used to model the entire dataset using Expectation - Maximisation. Since, this leads to a local optimal solution, the modelling is performed with different initialization to find the better fit. Bayesian Information Criterion is used to select the optimal number of Gaussians modelling the data. BIC calculates a tradeoff between the likelihood of the model and number of parameters used in the model.
Once, we have the model of the data, we predict the desired hitting speed, given the initial and the final desired position of the object using Gaussian Mixture Regression, which calculates desired speed as expectation of conditional probability $v_d = E(P(v|d, \theta))$. The desired hitting direction and the hitting speed are input to the Dynamical System that generates the hitting motion.

2.3.1.2 Limitations and Future perspectives

Although the learning system works well, it needs to be re-learnt for different types of boxes, such as those of different masses, different materials and shapes, and if hit by the robot in a different configuration. This requires the new data to be collected again which is quite a time consuming process without an automated data collection framework on real robots. We are developing an automated data collecting framework using the dual arm system that would require minimum human intervention and be able to generate data that spans different hitting configurations and different boxes. We are also working on understanding if a generalisation is possible and to what extent it is possible between the probabilistic

motion models of different boxes and different surfaces, which will help in reducing the amount of data needed from real experiments.

2.3.2 Publication: "Dual-arm control for coordinated fast grabbing and tossing of an object: Proposing a new approach"

2.3.2.1 Approach Summary

This work [2] proposes a solution based on dynamical systems (DS) to address the impact and tossing motion generation problem. 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. The proposed modulated dynamical system, at the position level, has the following form

$$\dot{\mathbf{x}}_d = M(\mathbf{x}) f_n(\mathbf{x}) + f_g(\mathbf{x}) \tag{2}$$

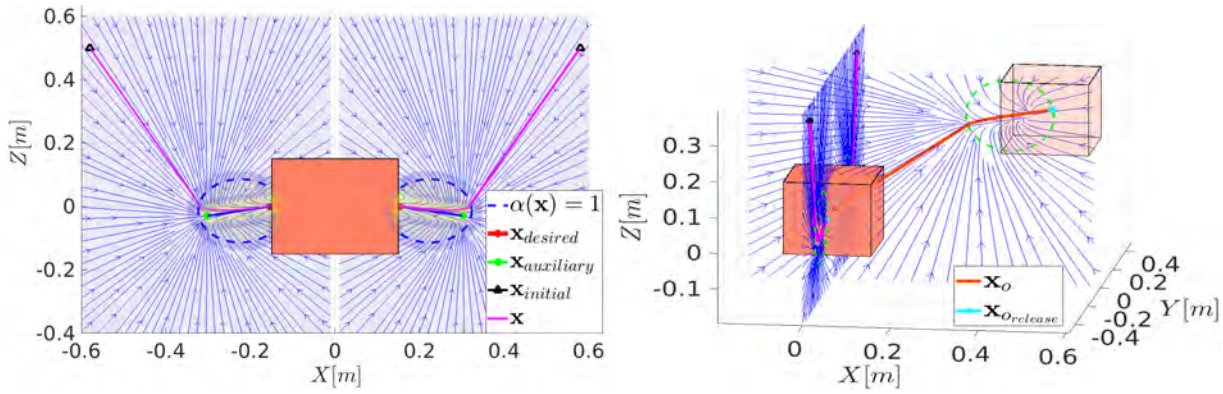
where, for a dual-arm robot, $\mathbf{x} = \begin{bmatrix} \mathbf{x}^L \\ \mathbf{x}^R \end{bmatrix} \in \mathbb{R}^6$ is the state vector of the DS with $\mathbf{x}^L$ and $\mathbf{x}^R$ representing the position of the left and right robot of the dual-arm, respectively. $f_n(\mathbf{x}) \in \mathbb{R}^6$ is the nominal DS that generates the coordinated motion towards transitory attractors located in the vicinity of the desired positions. $f_g(\mathbf{x})$ represents the equivalent grasping force in the motion space, whereas $M(\mathbf{x}) \in \mathbb{R}^{6\times 6}$ is the state-dependent modulation matrix that shapes locally the motion generated by $f_n(\mathbf{x})$. The modulation matrix $M(\mathbf{x})$ is defined such that

$$\dot{\mathbf{x}}_d = \begin{cases} f_n(\mathbf{x}) & \text{when non active} \\ f_m(\mathbf{x}) & \text{when active} \end{cases} \tag{3}$$

where the dynamics $f_m(\mathbf{x})$ is designed such that its attractor is: (I) sliding in the impact or tossing direction with the desired speed, and (II) fixed in all other directions. The grasping forces $f_g(\mathbf{x})$ are generated using Quadratic Programming (QP) to enforce the contacts constraints. Illustrations of the proposed DS motion flow when grabbing with impact and tossing are shown in Figure 14.

2.3.2.2 Main achievements, Limitations and Future perspectives

The modulated DS-based motion generation algorithm proposed in this publication [2] allows a dual-arm robotic system to quickly grab with impact and toss an object in one swipe. The desired states at contact and release are achieved through a local shaping of the robots' motions while preserving the coordination. The proposed DS-based motion generation algorithm, whose stability and convergence



(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.

towards the desired states were theoretically proved, has been experimentally validated in simulation and on real robots. An illustration of the grabbing and tossing task execution is provided through the snapshots of Figure 15.
The benefits obtained, in terms of task duration (cycle time) and energy expenditure, when using the proposed fast grabbing with impact and tossing in comparison to using the classical pick-and-place operation with near-zero contact and release velocities were systematically evaluated for different release velocities ranging from 0.5 m/s to 1.0 m/s. For each velocity set, we conducted five experiments of pick-and-place and five of pick-and-toss tasks.
An example of the robots' velocities resulting from the two approaches are shown respectively in Figure 16a (top) and (bottom), whereas the associated power and energy expenditure of both the left and right arm are shown in Figure 16b for the classical (top), and the proposed approach (bottom), respectively.
The overall results of this comparison are summarized in Figure 17a and Figure 17b, which show the average task duration and the energy expenditure of the two approaches, respectively.

The results confirmed that the proposed approach, besides motion coordination, enables to generate, for the dual-arm system, desired impact and tossing motions. The obtained results also suggest that grabbing with impact and tossing, especially when the impact direction anticipates the upcoming motion of the object, lead to shorter and more energy-efficient pick-and-place tasks. These results are consistent with those obtained in [9], where a pick-and-toss approach with a Delta robot was compared to a classical pick-and-place approach in a waste sorting facility.

However, in defining the parameters of the proposed dynamical systems, we used quasi-linear DS of the first order, which works well for fixed or slowly varying attractors but has no compensation ability in tracking and therefore defers this burden to the low-level torque controller. Moreover, the hand-tuned parameters do not generate optimal motions either in terms of energy or execution time.
Future work could use second-order DS to address the compensation problem in tracking as in [5]. In

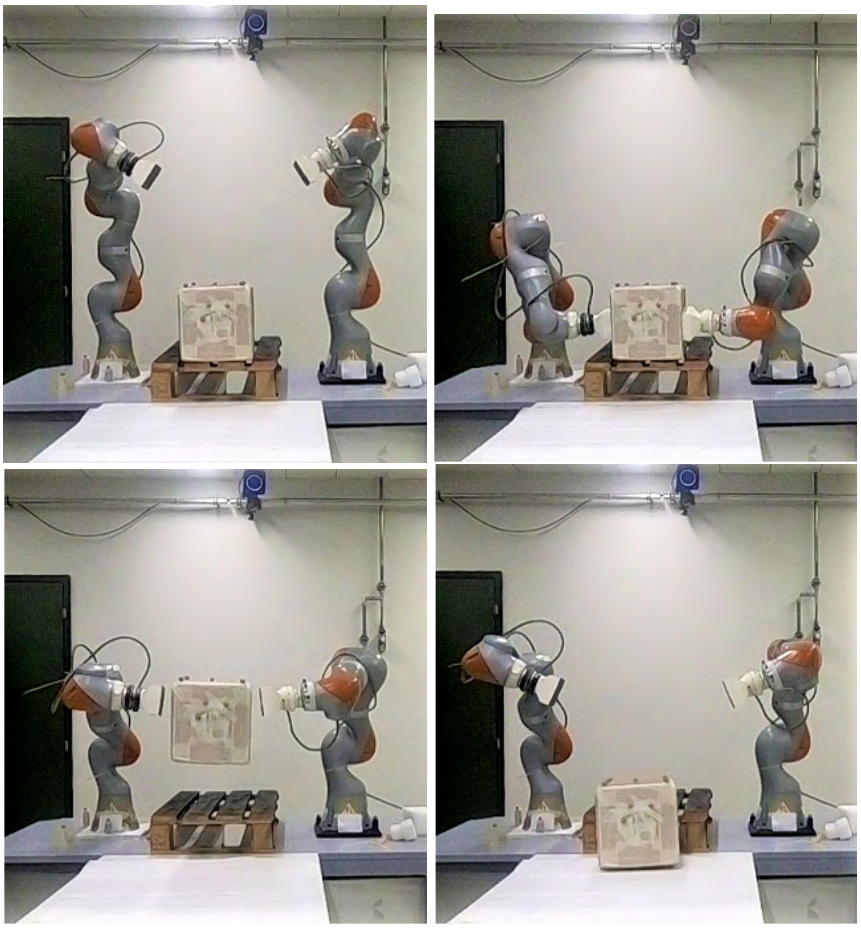


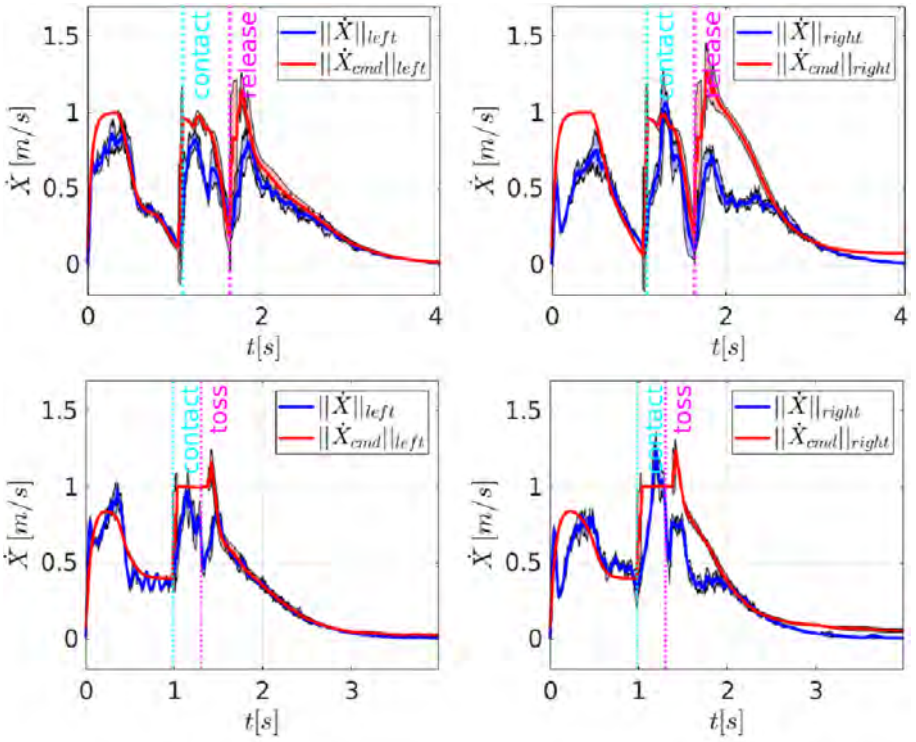
Figure 15: Snapshots of fast dual arm grabbing with impact and tossing an object. From left to right and top to bottom, the snapshots show: the initial robots' configuration, the pose of the robots' arm at initial contact with the object, robot arms tossing the object with the desired tossing velocity and at the desired release location; and fourth, the landing of the object.

addition, to address the optimality problem of the generated trajectories, one solution would be to use non-linear or Linear Parameters Varying (LPV) DS as in [6] and embed the optimality of the trajectories in their parameters. These DS parameters could then be learned from trajectories generated off-line by optimal controllers, for example, a minimum time controller (to minimize the cycle time), or a minimum energy control (to minimize the energy consumed during execution).

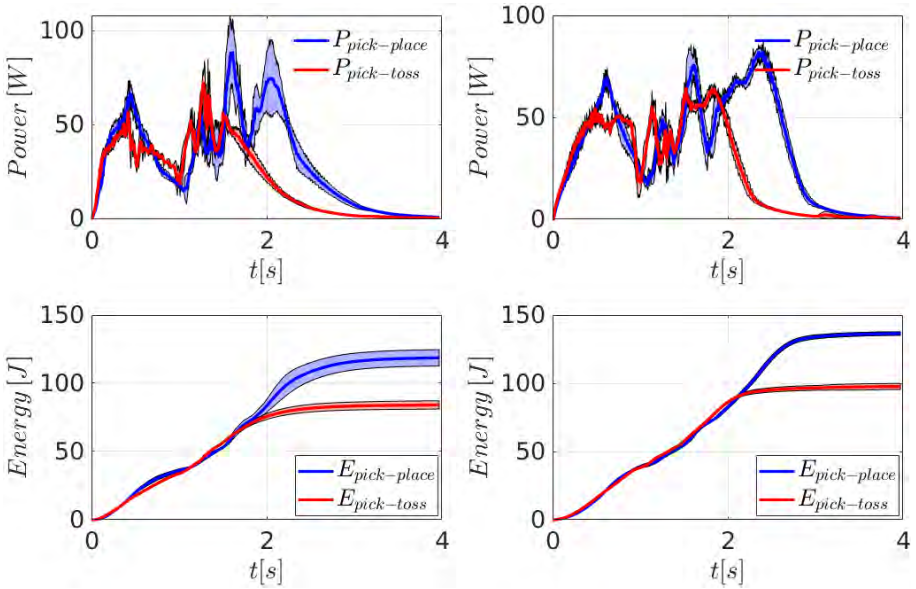
2.3.3 Publication: "Bimanual dynamic grabbing and tossing of objects onto a moving target"

2.3.3.1 Approach Summary

In this publication [3], the coordinated motion of the dual-arm system that picks up and tosses the object is generated by a modulated dynamical system as described in [2]. However, to ensure robust execution of the task under live perturbations in the speed or position of the moving target, we proposed as a complement to our previous DS an adaptation strategy to modulate the generated motion of the dual-arm system. The proposed adaptation strategy is twofold: first, a velocity modulation strategy, and

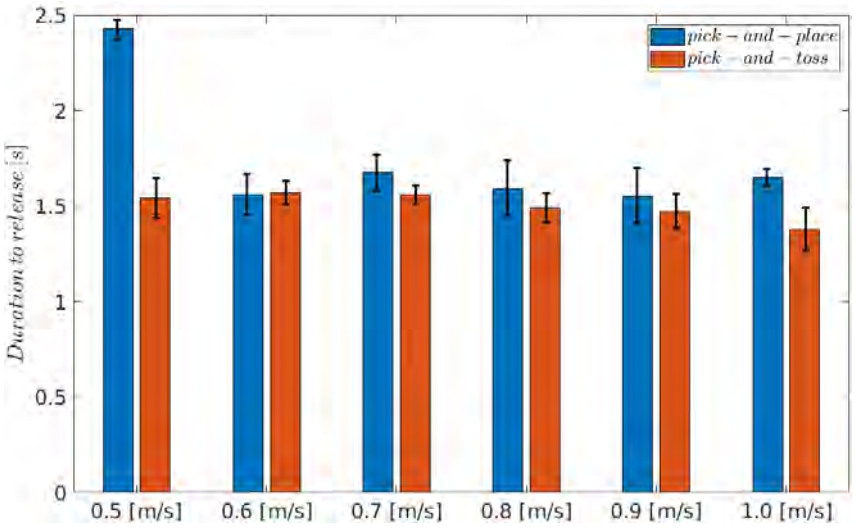


(a)

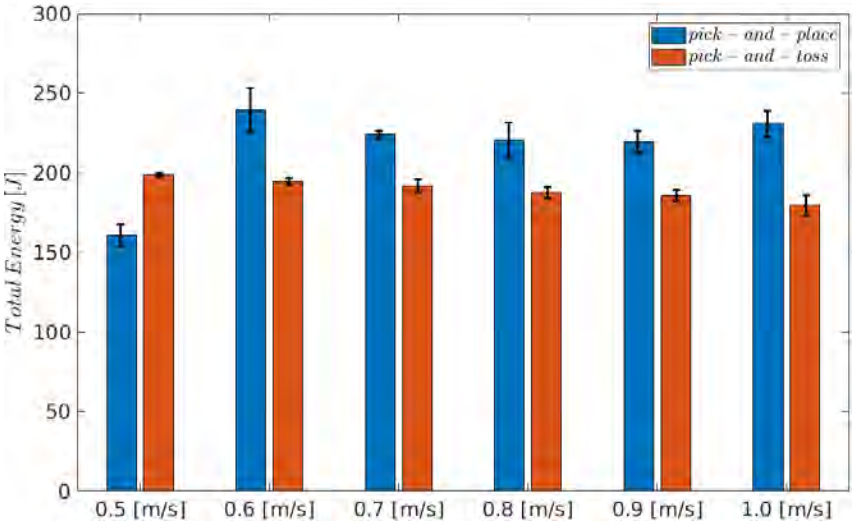


(b)

Figure 16: (a) Time evolution of mean and standard deviation of measured and commanded velocity norms of the left and right arm for five pick-and-placing (top), and five pick-and-tossing (bottom) experiments at 1.0 m/s. (b) Time evolution of estimated energy consumption of the left and right arm for five pick-and-placing (blue), and five pick-and-tossing (red) experiments at 1.0 m/s.



(a)



(b)

Figure 17: (a) Comparison for velocities ranging between 0.5 and 1.0 m/s of the task duration when the pick-up and placing happened at near-zero velocity (blue) and when the dual arm system leverage impact at the pickup and toss the object (red). (b) Comparison of overall energy expenditure of the dual arm system for tossing velocities ranging between 0.5 and $1.0m/s$ when grabbing the object with near-zero velocity (blue) and when performing fast object grabbing in a swipe (red).

second, an attractor adaptation strategy. At the velocity level, the DS can be accelerated or decelerated at will by multiplying the function by a positive scalar without affecting the stability properties at the attractor. Thus, the DS-based motion of the robot can be adapted to changes in the velocity of the moving target as follows

$$\dot{\mathbf{x}}_d = \beta(\mathbf{x}) M(\mathbf{x}) f_n(\mathbf{x}) + f_g(\mathbf{x}) \tag{4}$$

where $\beta(\mathbf{x})$ is the adaptation factor, which is simply a state-depend scaling factor.

In the second strategy, the attractor of the nominal DS $f_n(\mathbf{x})$ is adapted when the target's velocity changes its direction. This strategy introduces the possibility of reversing the robot's motion direction which is not achievable with the velocity modulation of a stable DS. Such reversal may be useful, for instance, to force the robot to retract toward an initial position. This allows the robot to have the momentum necessary to accelerate toward the desired throwing state without reaching the joint limits. Hence, we define the attractor as

$$\mathbf{x}_* = \alpha(\mathbf{x}^t, \dot{\mathbf{x}}^t)\mathbf{x}_d + (1 - \alpha(\mathbf{x}^t, \dot{\mathbf{x}}^t))\mathbf{x}_{stb} \tag{5}$$

where $\alpha(\mathbf{x}^t, \dot{\mathbf{x}}^t) \in [0, 1]$ is a target's state-dependent scalar function that goes to 1 or 0 depending on whether the target moves in the direction of the interception or not. $\mathbf{x}_d$ is the desired attractor of the nominal DS, and $\mathbf{x}_{stb}$ denotes a standby attractor to which the robot should retract to.

2.3.3.2 Main achievements, Limitations and Future perspectives

The adaptive modulated dynamical system proposed in [3] was validated experimentally on two 7-DoF robotic arms (KUKA LBR IIWWA7 and IIWA14) in tasks consisting of grabbing and tossing objects onto a target moving on a conveyor belt. Such tasks represent depalletizing scenarios commonly found in the industry. We assessed and demonstrated the accuracy, robustness, and adaptivity of the proposed algorithm to changes in the moving target's velocity. In other words, we showed how the proposed control strategy adapts the motion of the dual-arm system carrying the object for successful interception in the presence of changes in the target motion.
An illustration of the motion adaptation under live perturbations of the target's motion is provided through the snapshots of Figure 18 which shows successful interception of the target despite human-induced speed perturbations (target pulled back three times). The associated plots showing the position and velocity of the target–object–robot system under manually-induced perturbation and with adaptation to compensate for it are shown in Figure 19. The linear velocity norm of the left end-effector is shown on the top-right (the right end-effector is not shown, but follows a similar pattern); the adaptation factor $\beta(\mathbf{x})$ and the y-velocity of the target that drives it are shown at the bottom-right.

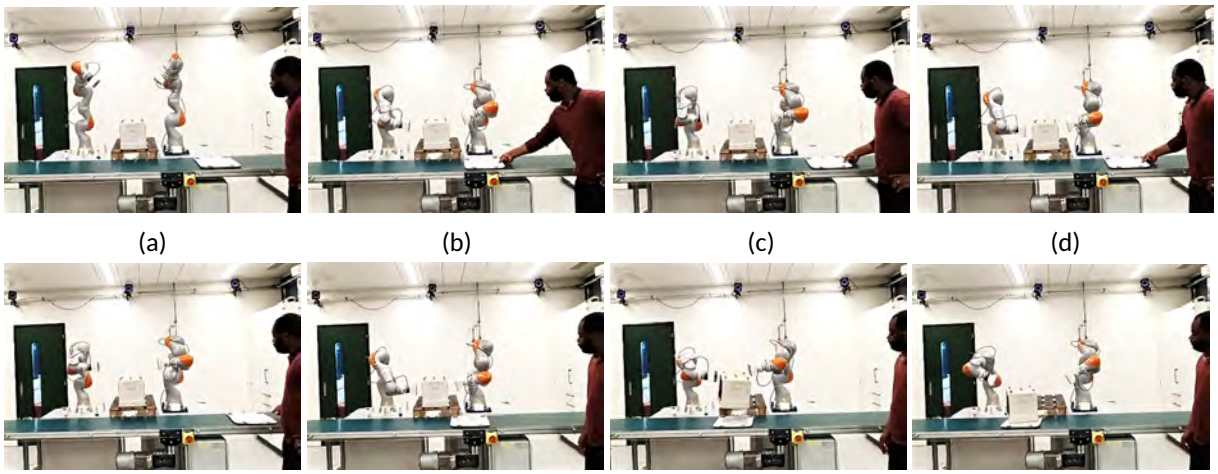


Figure 18: Snapshots illustrating adaptation of dual-arm system to manual perturbations of target motion in grabbing and tossing of an object onto a moving target: (a): dual-arm in standby, waiting for target to reach estimated state-to-go; (b)-(c) and (d)-(e): perturbation introduced by manually pulling back moving target, causing retraction of robots; (h)-(i): grabbing and tossing of object as target moves; (j): motion of object and target after successful interception. (See video: `https://youtu.be/8a4AFDYfrXo`)

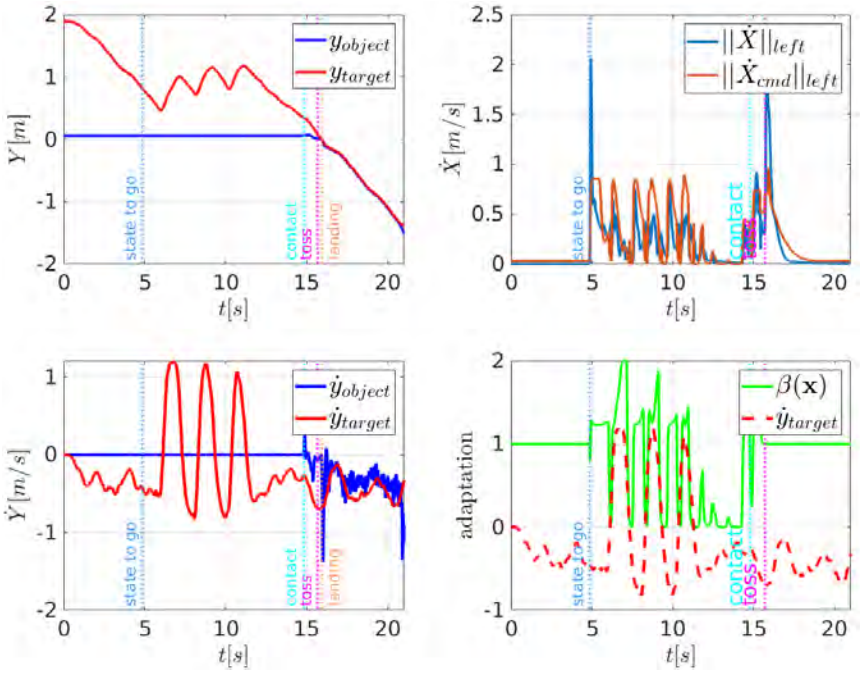


Figure 19: Illustration of position and velocity plots of the system with motion adaption while grabbing and tossing an object onto a moving target under manual perturbation: left: y-evolution of position (top) and velocity (bottom) of target (red) and object (blue), respectively, over time; (right): norm of linear velocity of left robot (top) and adaptation factor $\beta(\mathbf{x})$ and y-velocity of target (bottom). The robot motion was modulated based on estimated changes in target speed to ensure successful interception.

Although experimentally validated, the proposed DS-based dual-arm framework was only implemented in task space (where the grabbing constraints and the coordination mattered the most). Consequently, the DS is agnostic of what happens at the joint level and cannot take advantage of the robots' configurations to achieve a more efficient tossing motion.

Indeed, the motion modulation framework, at the core of the method presented in publication [2] and then used in [3], can be formulated both in task space and in joint space. It offers a very powerful framework to embed soft and hard constraints into dynamical systems. Similarly to obstacle avoidance constraints enforced through task space modulation, for instance in [7] or [8], one can also use joint space configuration-dependent functions in conjunction with the robot's kinematics to modulate the motion of the robotic system in order to satisfy, for instance, the robot joint limits or avoid self-collisions. Thus, such a solution will result in constraint-aware DS, capable of generating motions that are consistent with the constraints of a given robotic system as described in Deliverable 5.3 for the single-arm tossing task.

In publication [2] and publication [3], we generated desired impact velocities when grabbing objects without however taking into account the impact dynamics (both at the task level and at the control level). Throughout our experiments, the desired impact speeds were limited and the induced impact forces, as well as the jumps of velocities and torques, were assumed to remain within the safe limits for the robots and for the object. However, this assumption is limited since the impact forces and the velocity jumps do not only depend on the pre-impact velocities but also on the configurations of the robots and the properties of the impacting materials (e.g. rigidity, coefficient of restitution, etc.). To address the aforementioned problem, future works should include impact dynamics in determining impact velocity limits. Moreover, at the control level, a potential solution would be to combine the proposed DS with controllers such as those proposed in [10] or [11, 12], designed to keep force jumps and subsequent torque jumps within the limits of the robot. In addition, the continuity of the torques sent to the robots' motors can be ensured, for example, by using control strategies inspired by the reference spreading approach [13] or by projecting the command into an impact-invariant space as proposed in [14].

2.4 Ongoing Developments

2.4.1 Ongoing: "Learning impedance modulation for impact-aware manipulation"

While the task definitions of this project such as swiping and tossing, can define the motion and constrain the solutions space of the forces, each of these actions also entails uncertainty. Models of uncertainty at impact used in T.2.1 from T1.4 must be embedded to create inherently robust control laws. To this end, we will exploit impedance control laws driven by dynamical systems to embed models of feasible motions. In such a model, the impedance represents an envelope around the trajectories of the dynamical system. In this region, all trajectories lead to a feasible solution to the task.

The goal of the work is to estimate the damping matrix used for the dual arm tossing task such that the task is completed robustly and as fast as possible. This work builds upon the results presented in Publication 2.2.2, Publication 2.2.3, Publication 2.3.2, and Publication 2.3.3. The previous works employed a passive dynamical systems approach of the form,

$$\boldsymbol{F} = -D(\boldsymbol{x})\Big(\dot{\boldsymbol{x}} - f(\boldsymbol{x}) - D(\boldsymbol{x})^{-1}\boldsymbol{f}_e^d\Big) \tag{6}$$

Where $\boldsymbol{F}$ is the force command sent to the robot, $\boldsymbol{q}$ is the joint space position, $\boldsymbol{x}$ is the robot end-effector

position, $D(\boldsymbol{x})$ is the configuration dependent damping matrix to be learned, $f(\boldsymbol{x})$ is the dynamical system to be learned, and $\boldsymbol{f}_e^D$ is the desired end-effector force required for force closure. To move optimally fast prior to contact, we will first plan in joint space and learn the damping matrix. The learning will be accomplished in three steps:

Step 1: Determine Contact State: The end-effector position $\boldsymbol{x}_d^f$ and velocity $\dot{\boldsymbol{x}}_d^f$ are defined by the trajectory required to accelerate the box. However, for redundant robots such as the KUKA LBR IIWA and the FRANKA EMIKA which have 7 degrees of freedom, the joint space configuration $\boldsymbol{q}_d^f$ and joint space velocity $\dot{\boldsymbol{q}}_d^f$ at the time of contact with the box is not uniquely defined. In this way, the redundancy of the robot can be exploited to select for preferable inertia and kinematic stiffness. The goal of this task is to minimize the absolute time of the task. Thus, to achieve the fastest impacts with the smallest forces and kinematic stiffness normal to the box will be minimized at the time of impact. We will search for or learn the nominal minimum inertia and stiffness offline. This approach will be similar to that of the 'velocity hedgehog' approach introduced by Liu et al. [15]. This type of learning will entail sampling a large number of joint configurations. Then the index of the configurations that correspond the to lowest stiffness and flux in a particular direction will be determined.

Step 2: Compute Dynamical System: Given the initial joint configuration, final positions of joint configuration, the joint position limits, the joint velocity limits, and torque limits, polyMPC[16] can be used to generate time optimal trajectories from many starting states to a final state. Using the time optimal trajectories, a joint space dynamical system, denoted $f(q)$, will be learned. In this way, the learned dynamics system will be an approximation of the time-optimal solution constrained by the physical limitations of the robot.

Step 3: Compute Impedance: Given $f(\boldsymbol{q})$ the goal is to determine the optimal parameters for $D(q)$. As a first step, we will solve for a diagonal impedance matrix with a constant coefficient. The cost to be minimized will be the potential and kinetic energy of the robot system subject to the constraints that the terminal joint position and joint velocity are achieved.

2.5 List of I.Learn Softwares

2.5.1 Single-arm controller for hitting an object

Two control strategies were derived from the solution presented in publication "Learning to Hit: A statistical Dynamical System based approach" (cf. section 2.3.1): one based on velocity control and the other relying on torque control. Both controllers enable a robot to perform a hitting task on an object with a desired hitting speed and hitting direction.

The first implementation can be accessed through this Git repository `https://github.com/epfl-lasa/hitting_sim/tree/main`. This controller is based on velocity control. It was implemented in Python, primarily using Pybullet as simulator before integrating the Algoryx simulator.

The second implementation is available in this Git repository `https://github.com/epfl-lasa/i_am_project` . The library is written in C++ and is integrated with Robot Operating System (ROS) providing a versatile platform for application in both physical and simulated environments. It currently runs with the Gazebo simulator and on the KUKA LBR iiwa 7 robot.

2.5.2 Dual-arm controller for grabbing and tossing an object

The codes associated with the publication ”Dual-arm control for coordinated fast grabbing and tossing of an object: Proposing a new approach” (cf. section 2.3.2) and the publication ”Bimanual dynamic grabbing and tossing of objects onto a moving target” (cf. sections 2.2.3 and **??**) can be found on the following GitHub repository: `https://github.com/epfl-lasa/iam_dual_arm_control`. This repository contains codes to generate coordinated motion and forces to control a robotic dual-arm system to grab and toss objects on fixed and moving targets within the tossable space of the dual-arm system.

3 CONCLUSION

In this Deliverable D2.2, we provided an overview of the publications associated with the I.Learn framework. The report focused essentially on tasks T2.2 and T2.3 and provided approach summaries and main results related to each publication pertaining to each task. It also discussed the limitations of the current achievements and provided prospective directions. More specifically, the report presented the main methods and results on the determination of impact postures and the generation of impact and throwing motions. In general, the solutions described in the report and tested on real robots largely meet the requirements of tasks T2.2 and T2.3. However, all the presented approaches explicitly or implicitly assume kinematic feasibility, which does not guarantee the dynamic feasibility of the considered tasks. The latter, which is more complex, is much more general because, beyond kinematics, it takes into account the inertial properties of robots in motion as well as their initial conditions. Current developments to be completed in the next six remaining months of the project are attempting to address these limitations by learning dynamic constraints in conjunction with impedance associated with the tasks, as part of Task 2.4 (ongoing). The final solutions of I.Learn will be integrated with those of I.Model, I.Sense and I.Control and then validated in the GRAB scenario, which will be reported in Deliverable 5.5.

4 REFERENCES

[1] H. Khurana, M. Bombile and A. Billard, "Learning to Hit: A statistical Dynamical System based approach," 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 2021, pp. 9415-9421, doi: 10.1109/IROS51168.2021.9635976.

[2] M. Bombile, A. Billard, Dual-arm control for coordinated fast grabbing and tossing of an object: Proposing a new approach, IEEE Robotics Automation Magazine 29 (3) (2022) 127–138. doi:10.1109/MRA.2022.3177355.

[3] M. Bombile, A. Billard, Bimanual dynamic grabbing and tossing of objects onto a moving target, Robotics and Autonomous Systems, (2023), 104481, ISSN 0921-8890, https://doi.org/10.1016/j.robot.2023.104481. (https://www.sciencedirect.com/science/article/pii/S0921889023001203)

[4] H. Khurana, A. Billard, Motion Planning and Inertia Based Control for Impact Aware Manipulation, IEEE Transactions of Robotics (T-RO) (2023 - *Accepted*)

[5] S. S. Mirrazavi Salehian, M. Khoramshahi, and A. Billard. A dynamical system approach for catching softly a flying object: Theory and experiment. in IEEE Transactions on Robotics, vol. 32, no. 2, pp. 462-471, April 2016., 2016.

[6] S. S. Salehian Mirrazavi, Nadia Figueroa and Aude Billard, Dynamical system-based motion planning for multi-arm systems: Reaching for moving objects. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, (2017), (pp. 4914-4918)

[7] Seyed Mohammad Khansari-Zadeh and Aude Billard. A dynamical system approach to realtime obstacle avoidance. Autonomous Robots, 32(4):433-454, 2012.

[8] L. Huber, A. Billard, and J.-J. Slotine. Avoidance of convex and concave obstacles with convergence ensured through contraction. IEEE Robotics and Automation Letters, 2019.

[9] F. Raptopoulos, M. Koskinopoulou, and M. Maniadakis. Robotic pick-and-toss facilitates urban waste sorting. In 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), pages 1149-1154. IEEE, 2020.

[10] Y. Wang, Niels Dehio, A. Tanguy, and A. Kheddar. Impact-Aware Task-Space Quadratic-Programming Control. working paper or preprint, November 2020. URL https://hal.archives-ouvertes.fr/ hal-02741682.

[11] N. Dehio and A. Kheddar. Robot-Safe Impacts with Soft Contacts Based on Learned Deformations. In ICRA, Xi'an, China, May 2021. URL https://hal.archives-ouvertes.fr/hal-02973947.

[12] N. Dehio, Y. Wang, and A. Kheddar. Dual-arm box grabbing with impact-aware mpc utilizing soft deformable end-effector pads. IEEE Robotics and Automation Letters, 7(2):5647-5654, 2022.

[13] M. Rijnen, A. Saccon, and H. Nijmeijer. Reference spreading: Tracking performance for impact trajectories of a 1dof setup. IEEE Transactions on Control Systems Technology, 28 (3):1124-1131, 2019.

[14] W. Yang and M. Posa. Impact invariant control with applications to bipedal locomotion. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5151-5158. IEEE, 2021.

[15] Y. Liu, A. Nayak, and A. Billard. A solution to adaptive mobile manipulator throwing. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1625–1632. IEEE, 2022.

[16] P. Listov, and C. Jones. Polympc: An efficient and extensible tool for real-time nonlinear model predictive tracking and path following for fast mechatronic systems. Optimal Control Applications and Methods, 41(2):709–727, 2020.