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

Scenario 2 (BOX) report

Dissemination level	Public (PU)
Work package	WP5: Integration and Scenario Validations
Deliverable number	D5.4
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**
Alexander Oliva	TU Eindhoven	a.a.oliva@tue.nl
Fredrik Nordfeldth	Algoryx Simulation AB	fredrik.nordfeldth@algoryx.com
Maarten Jongeneel	TU Eindhoven	m.j.jongeneel@tue.nl
Ricardo Duarte	Smart Robotics	rduarte@smart-robotics.nl
Steven Eisinger	Smart Robotics	seisinger@smart-robotics.nl

Control sheet

Version history			
Version	**Date**	**Modified by**	**Summary of changes**
0.1	30/12/2023	Rico & Steven	TOC & first contents
0.11	28/02/2024	Fredrik Nordfeldth	Started with Simulation section
0.12	6/03/2024	Maarten Jongeneel	Restructured Document
0.13	7/03/2024	Maarten Jongeneel	Start on Sections 4.1-4.4
0.14	8/03/2024	Rico & Steven	Work on Chapter 1, 3, and 5
0.17	15/03/2024	Maarten Jongeneel	Finalizing Chapter 2 and Chapter 4
0.18	18/03/2024	Alexander Oliva	Work on Chapter 2
0.19	19/03/2024	Alexander Oliva	Work on Chapter 4
0.50	13/03/2024	Maarten Jongeneel	Pre-final version ready for peer-review
0.51	18/03/2024	Alessandro Saccon	Pre-final version with comments by Reviewer 1
0.52	26/03/2024	Rico & Steven	Addressed comments from peer-review
0.55	28/03/2024	Alexander Oliva	Pre-final version: finalizing Chapter 4, added appendix C, added tables RMSEs
0.9	29/03/2024	Steven Eisinger	Peer-review comments addressed
1.0	04/04/2024	Jos den Ouden, Alessandro Saccon	Revised version ready for submission, quality check

Peer reviewers		
	Reviewer name	**Date**
Reviewer 1	Alessandro Saccon	04/04/2024
Reviewer 2	Ali Baradaran	20/03/2024

Legal disclaimer

Executive summary

Industrial robotics applications that dynamically interact with their environment tend to actively avoid collisions with it. While this is reasonable for the most part, some performance is lost due to margins being taken around objects, which potentially slow the robot down by forcing it to move in sub-optimal and slightly longer paths, or limit the amount of items that can be fit inside a container. Furthermore, the tighter the working area of a robot is, the bigger the risk of collisions with the environment and, subsequently, the bigger the impact on robot performance. However, *some collisions* could in theory be tolerated by the robot. For example, it might be reasonable to allow the robot to collide (or contact) with movable or soft objects - especially if the outcome of such a collision can be predicted and the performance loss due to conservative collision avoidance minimized.

To achieve some form of usable collision tolerance (contact-rich interaction) in a robotics application, better online planning as well as accurate robot-environment modeling and better visual-tactile perception are needed. While planning is not a core technology within the H2020 I.AM. project, the goal of the BOX scenario is to demonstrate that if sufficiently accurate modeling and numerical simulations of contact-rich interactions are achievable, the outcome of these contacts and collisions with the environment can be reliably predicted. Combining this accurate modeling with sufficiently sophisticated vision perception and planning algorithms, the robot would be able to autonomously execute and adapt item packing strategies which actively explore environmental contact. This opens the possibility for further research opportunities into such algorithms.

One application of this technology is with industrial item picking (IP) robots. IP robots pack mixed item types into containers before being shipped to other destinations. The lack of contact-rich motion planning results in sub-optimal container packing when compared to what a human operator can achieve, leading to less effective use of space and fewer items per shipment to distribution centers or stores, which has an adverse effect on transportation costs. Through better forward simulation and allowing items to collide with each other, margins between items can be reduced (or even eliminated), resulting in a denser packing of containers in many cases. This motivates the use of an IP setup for the BOX demonstrator. Components for BOX were developed in order to validate this hypothesis and provide accurate environment simulation to enable contact-rich motion planning.

When items are picked at locations other than their center of mass, they will rotate, creating an opportunity for collisions with adjacent items during placement in a container. To enable placing items in containers with smaller margins, the dynamics of the gripper and the geometry and mass distribution of the picked item must be *sufficiently* known in order to predict their behavior during and after contact with the environment and other items. One of the goals of the BOX scenario is to understand how sufficiently known these properties need to be in order to achieve contact-rich packing.

Exploiting the symmetry in suction cup grippers for industrial item picking robots, a simplified dynamic model was developed for the suction cup. If the inertial properties of a picked item, the suction cup gripper, and the robot are sufficiently known, the item's orientation can be predicted at placement time in a repeatable manner, allowing for smaller margins between adjacent items with less risk of damage during placement motions.

Item, suction cup, and test setup models and properties have been integrated using the RACK framework, allowing forward simulation of the item's orientation during the placement motion.

To show the effectiveness of the developed models for predicting item orientation and interaction forces,

a test setup consisting of an item picking robot (UR10), depth camera (Zivid), motion tracking system (OptiTrack), force-torque sensor (BOTA SenseONE), and a target container to place items in is constructed to collect data on item motions. Items have been created and their inertial measurements taken and modeled. The motion planning was done by manually programming motions. Packing patterns were also created manually. A comparison between the measured and simulated item orientations is conducted.

The developed models are validated to reflect reality with impressive levels of accuracy. Combining the I.AM.-developed simulation environment with (at the moment nonexistent) sufficiently intelligent motion and packing algorithms could yield packing density performance (filling degree) far superior to the current state of the art and comparable to that of a human operator. This encouraging result provides motivation for further research into contact-rich motion planning, and packing algorithms that take a contact-rich environment into account. In the long-run, this may facilitate the adoption of industrial robotic applications in the e-commerce industry.

TABLE OF CONTENTS

ABBREVIATIONS

Abbreviation	Definition
CAD	Computer-Aided Design
DH	Denavit-Hartenberg
EC	European Commission
FSM	Finite State Machine
I.AM.	Impact-Aware Manipulation
ID	Identity
IP	Item Picking
KPI	Key Performance Indicator
LSPB	Linear Segments with Parabolic Blends
MDF	Medium-Density Fiberboard
PU	Public
RD	Random Dimension
RGB	Red-Green-Blue
SRIP	Smart Robotics Item Picker
TCP	Tool Control Point
TOC	Table of Contents
TRL	Technology Readiness Level
TU/e	Eindhoven University of Technology
TUM	Technical University of Munich
URDF	Unified Robot Description Format
WP	Work Package

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 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 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. potentially enables shorter cycle time and increased packing efficiency 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 demonstrating object fast boxing, and a case depalletizing scenario demonstrating object grabbing.

1.2 BOX scenario background

For the BOX scenario we are aiming to improve the process of current bin-to-bin applications, where the robot transfers items from one container to another container. In most state-of-the-art bin-to-bin applications, collisions between the item being placed and other items inside the target container are usually undesirable. For this reason, the item has to be placed within a certain margin from other items to account for the error in the detection of the item size - usually due to uncertainties introduced by image acquisition and computer vision processing algorithms.

With the BOX scenario, however, the item and the robot are allowed to interact (i.e. collide) with other items and the environment. If we can let items collide with one another and can predict the pose of the contents of the container after collision, we can use this to place items closer to each other and fill the container more effectively (see Figure 1 for an illustration of the scenario). In order to exploit collisions for effective packing, models of the target container, the items therein, the robotic arm, the gripper, and the item to be placed must be known in order to predict their collision behavior. Therefore, the BOX scenario aims to model and describe the dynamic components of the bin-to-bin system such that the robot may place an item into the target container in such a way that the item can contact with another without harming the items, the robot, or the container.

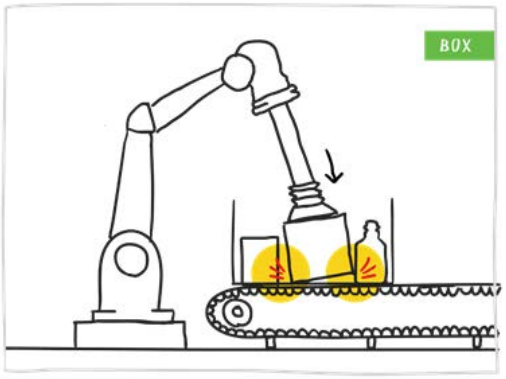


Figure 1: BOX scenario

1.3 Overview of current industry standards

Within e-commerce distribution centers, human workers are tasked with moving items (boxes, bags, etc.) to endpoints such as bins, pallets and conveyor belts. Demand for robotic automation solutions is increasing due to increased labor scarcity and the need for more future-proof working environments. As companies transition from human-focused to robot-focused solutions, industrial robots typically have to deal with less structured environments (which were originally designed for human workers) and a high variation in the products to handle. Currently available robot solutions are often slower, less reliable, and pack items into containers less effectively than humans, which is slowing down industrial robot adoption due to increased transportation and operation costs [1].

E-commerce distributors (which have boxed or bagged items in their in-feed) must pack containers which are eventually shipped to other distribution centers or stores. When human workers pack these containers, they have the ability to reorient, redistribute, and press items against each other to achieve dense packing. Current state-of-the-art robots take measures to avoid collisions and allow consistent placing behavior, such as using large margins between placed items. Allowing items to touch or press against each other while a robot is placing an item has the potential to improve container density by reducing or eliminating these margins. More densely packed containers mean more densely packed trailers, resulting in reduced shipping cost per item as fewer (or smaller) trailers may be needed to transport the same amount of items [1]. Consequently, the lower total fuel cost translates to a lower carbon footprint compared to loosely packed containers, as less fuel is consumed in shipping.

Hence, the main driving key performance indicator (KPI) for the BOX scenario is how densely packed a container is, which we call **filling degree** within the context of this report. Further elaboration on how this is calculated is found in Section 1.4.

Further information about the business case for BOX can be found in Deliverable D7.3 [1].

1.4 Filling Degree

When placing items inside a container using a robotic manipulator, collisions between items or the robot and environment are avoided as they may result in item, robot, or environment damage, which may include sensors, cages, and other expensive equipment. To make sure items do not collide with one another, they have to be placed a comfortable margin apart from each other, to offset uncertainties introduced by sensors and placing algorithms.

However, in the BOX scenario, collisions are allowed between items and the robot (contact-rich approach). These explicit contacts will be allowed such that items can be placed closer together. Compared to currently deployed systems, this should allow a robotic system fitted with I.AM.-developed technology to pack more items in a given container with smaller margins between items.

In order to quantitatively evaluate the ability of a system to effectively pack items into a container, the **filling degree** metric is devised in the context of the BOX scenario. This metric provides a way of comparing the item packing performance of the BOX demonstrator relative to the state of the art and manual operator.

The **filling degree** is defined as the ratio between the volume occupied by the items placed inside the container and the total inner volume of the container. It is important to note that the filling degree depends on both the shape and volume of the items and the algorithm that determines the place pose of the item. When comparing two systems, a higher filling degree attained by a certain system on a certain container indicates that system is able to pack more items inside said container than the system it is being compared to. Hence, in a production context, an improvement in filling degree might lead to less frequent container switches and, consequently, savings in production time. Additionally, containers for which a higher filling degree is achieved use space more effectively, since a container always takes the same amount of space regardless of how full it is. Shipping these containers may become more efficient, as more items can be transported per trailer, potentially reducing transportation costs.

The filling degree can be calculated as follows:

$$filling\ degree = \frac{100\%}{v_{container}} \sum_{i=0}^{n} v_i \qquad (1)$$

where n is the number of items packed in the container, i represents an item index, v_i is the volume of item i, and $v_{container}$ is the internal volume of the container.

This performance indicator will form the basis of evaluating the performance of the BOX demonstrator relative to the state of the art and manual operators' performance.

1.5 Purpose of the deliverable

This deliverable aims to provide insight on the demonstration of the BOX scenario as well as an overview of the components developed by the I.AM. consortium over the course of the project relevant to this scenario and corresponding performance testing to show the results of the BOX scenario. Included are the preparation and steps performed in programming, testing, and evaluating a robotic arm at the Vanderlande Innovation Lab at TU/e in Eindhoven for the BOX scenario. As part of the I.AM. BOX scenario, a demonstrator of TRL (technology readiness level) 5 was created to both demonstrate and evaluate the industrial potential of I.AM. developments in the context of the BOX scenario. The methodologies applied to prepare the demonstrator are also part of this deliverable.

The BOX scenario builds on the RACK framework developed for the I.AM. project, the mc_rtc framework developed by CNRS, the AGX Dynamics simulator developed by Algoryx, models developed at TU/e, and business value and data provided by both Smart Robotics and Vanderlande. Through the combination of these components and testing through the BOX demonstrator, a simulation environment is presented which predicts the position and orientation of boxes during flipping and placement into a container as deemed relevant by Smart Robotics and Vanderlande for the e-commerce industry. Through the comparison of this simulation environment with reality, a basis is formed for further research into contact-rich motion planning.

1.6 Intended audience

The dissemination level of this report is 'public' (PU) - meant for members of the Consortium (including Commission Services) and the general public.

2 Developed Components for BOX scenario

In this section, one can find the various components that were developed in order to validate the concept of contact-rich boxing. The architecture is discussed, giving an overview of the system components and how they relate to each other. Secondly, each component is individually discussed. These components are (a) the suction cup model, (b) its integration into the AGX Dynamics physics simulator, and (c) the controller tasked to execute the planned BOX motion (mc_rtc).

2.1 Suction cup modeling

One of the most important technological advancements that contributed to the realization of this scenario is the previous development and validation of the suction cup model during the holding phase and its subsequent implementation in the AGX dynamics simulator for this deliverable. Preliminary results on the suction cup modeling and identification were presented in deliverable D1.3 [2] and the complete results are under review for possible publication to IEEE Transactions on Robotics [3]. In this paper, a 6D force-displacement model was validated and allows us to accurately predict the suction cup's deformation during the holding phase while carrying an object (of known mass, center of mass location and inertia) as well as the interaction forces involved. In principle, the inertial parameters of the payload can be estimated online, demonstrated in previous work by by TUM [4] using the momentum observer for tactile robots. The extension of the work to online payload estimation method for the flexible links is planned for future works. Therefore, in this deliverable we are assuming the inertial parameters of the payload to be known.

A schematic overview of the interaction forces acting on the system can be seen in the free body diagram of Figure 2. The wrench acting on the package applied by the suction cup has been modeled as a driven spring and a damper, hence, two effects are acting from the suction cup on the package as

$$ {}_S\mathbf{f}_{SC\to PA} = {}_S(\mathbf{f}_{SPRG})_{SC\to PA}({}^E\mathbf{H}_S;\mathbf{K}) + {}_S(\mathbf{f}_{DAMP})_{SC\to PA}({}^E\mathbf{H}_S;{}^S\mathbf{v}_{E,S};\mathbf{D}) \tag{2} $$

with $\mathbf{K},\mathbf{D}\in\mathbb{R}^{6\times 6}$ the symmetric and positive (semi-)definite stiffness and damping matrices, respectively, and ${}^S\mathbf{v}_{E,S}$ the twist between frames E and S. The spring wrench has been derived from a newly proposed geometric potential energy function defined on all $\mathrm{SE}(3)$, and has the following expression

$$ {}_S(\mathbf{f}_{SPRG})_{SC\to PA} = -\frac{1}{4}\begin{bmatrix} {}^S\mathbf{R}_{S_1} & \mathbf{0}_{3\times 3} \\ {}^S\mathbf{o}_{S_1}^{\wedge}\,{}^S\mathbf{R}_{S_1} & {}^S\mathbf{R}_{S_1}\end{bmatrix}\begin{bmatrix}\mathbf{R}^\top+\mathbf{I} & \mathbf{0}_{3\times 3} \\ -(\mathbf{R}^\top\mathbf{o})^\wedge & (tr(\mathbf{R})\mathbf{I}-\mathbf{R})\end{bmatrix}\mathbf{K}\begin{bmatrix}(\mathbf{R}^\top+\mathbf{I})\mathbf{o} \\ (\mathbf{R}-\mathbf{R}^\top)^\vee\end{bmatrix}, \tag{3} $$

where we have used $\mathbf{H}=(\mathbf{R},\mathbf{o})$ to indicate the local deformation ${}^{S_2}\mathbf{H}_{S_1}=({}^{S_2}\mathbf{R}_{S_1},{}^{S_2}\mathbf{o}_{S_1})$.

A thorough investigation of the symmetry properties of the spring wrench model was conducted leading to a simpler structure of the stiffness matrix and the identification procedure. The numerical values of the stiffness matrix have been identified, and the model validated, from a dataset containing 1200 static poses carrying objects with different masses and center of mass locations.

As an indication of the achievable pose prediction in steady state using the identified stiffness parameters, for an object of about 1.75 Kg, we obtain a pose error in the order of 5 mm and 3°, with a gripper inclination of 60°. The results of a quasi-static experiment carrying a payload of 1.9 Kg is shown in Figure 3. Starting from a straight vertical gripper configuration, the robot inclines and holds the gripper in place for a few seconds, to allow for steady state to be reached. This process is repeated for different inclination angles, before returning step-by-step to the initial configuration.

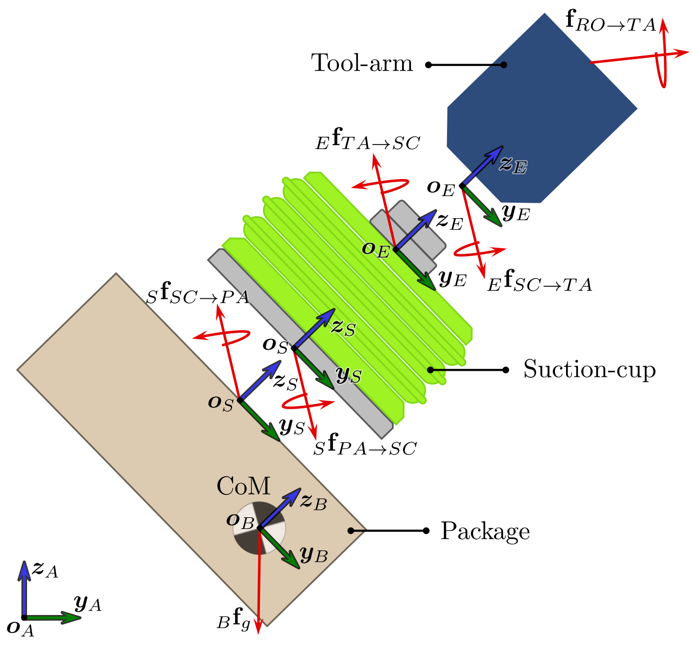


Figure 2: Free Body Diagram of the bellows suction cup and package in a planar perspective. In the picture, ${}_B\mathbf{f}_g$ is the gravitational wrench acting on the center of mass frame of the package. ${}_S\mathbf{f}_{SC\to PA}$ is the wrench acting on the package applied by the suction cup, and similarly for the other expressions.

2.2 Physics simulator integration

The simulation environment model, including robot, conveyor belt, boxes, and containers are defined using the BRICK modeling language. BRICK is described in detail in Deliverable D1.2 of the I.AM. project [5]. The RACK [1] framework (previously known as GLUE) developed during I.AM. integrates the QP robot control framework mc_rtc developed by CNRS partner through the open scene description format CLICK [2] developed by the I.AM. partner Algoryx, which makes it possible to run the exact same controller implementation for both the real setup and the simulation environment. RACK is integrated with AGX Dynamics [3], which has been previously validated during I.AM. for impacts [6].

2.2.1 Suction cup validation

Using BRICK the suction cup is extended to be flexible and validated to perform as the real cup. The suction cup model described in 2.1 for the holding face is implemented in an AGX Dynamics python callback in RACK. The stiffness[4] and mechanical damping[5] parameters for the simulated suction cup are identified by validating the simulation with recorded real world data. 1200 different static configurations with different boxes attached to the gripper at different angles are recorded to identify the stiffness parameters. To identify the damping parameters the motion of the attached box is recorded when displaced and re-

[1]https://gitlab.tue.nl/robotics-lab-public/glue-application.git
[2]https://github.com/Algoryx/click-mirror.git
[3]https://www.algoryx.se/agx-dynamics/
[4]$[\frac{Nms}{rad}]$ for rotational dimensions and $[\frac{Ns}{m}]$ for linear dimensions
[5]$[\frac{Nm}{rad}]$ for rotational dimensions and $[\frac{N}{m}]$ for linear dimensions

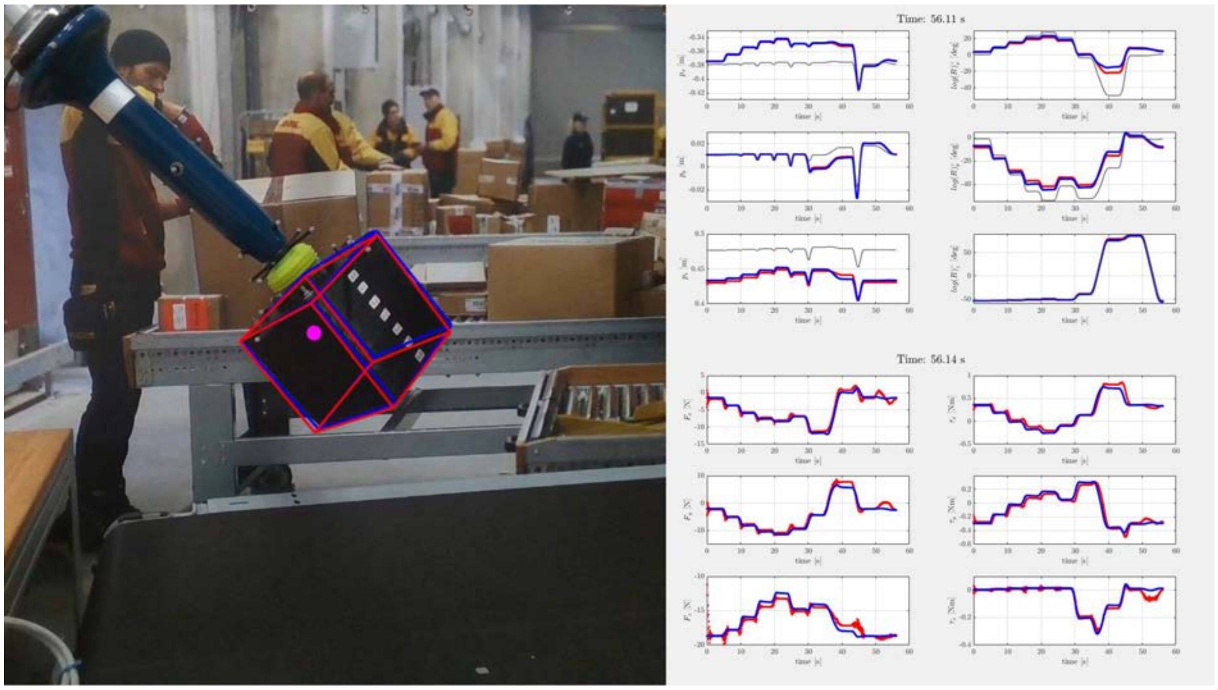


Figure 3: Suction cup model prediction results: On the left figure, a video frame (at 27 seconds) showing a vacuum gripper tilted about $50°$ during a quasi-static experiment. It is equipped with a single suction cup and is holding a payload with eccentric mass of $1,9$ Kg (●). The box's model is projected on the image using both ground-truth (—) and the predicted (—) pose data. On the right, 12 plots showing the three position (top left) and rotation vector (top right) components of the box's pose as well as the force (bottom left) and torque (bottom right) components of the gravitational wrench acting on the suction cup. For both poses and wrenches, the ground-truth (—) and predicted (—) data is shown. The pose of the gripper's end-point is also reported (—).

leased to reach a static state. For the validation, an application called *ClickOptimize* is developed within RACK. It uses *scipy optimize* [6] to minimize the error in position and angle for the simulated trajectory. *ClickOptimize* requires a BRICK model that extends *OptimizeSceneCSV* in the CoreOptimizeSceneCSV.yml file.

2.3 Controller Design

The mc_rtc [7] framework is used to control both the real robot and the simulated robot in AGX Dynamics. It has been developed by CNRS, and further developed within the I.AM. project to support the control of motions with impacts. Details of this development are also discussed in Deliverable D4.1 [8]. With the RACK framework as briefly described in 2.2 and more extensively in Deliverable D1.2 [5], the mc_rtc controller can be used both on the real robot as well as on the simulated robot without the need to change the controller, only requiring some configuration files specific to the setup used. This workflow is represented in Figure 4. Within the mc_rtc framework, we choose to use a Finite State Machine (FSM) to set up the controller. This framework allows us to create a single controller that can be executed on

[6]https://docs.scipy.org/doc/scipy/reference/optimize.html

Attribute	Type	Purpose
optimize_position	Bool	Optimize function will minimize error of distance
optimize_rotation	Bool	Optimize function will minimize error of relative quaternion distance
rot_pos_error_ratio	Real	Weight between error of rotation and position
only_final_state	Bool	True, error relative last frame. False, error for full trajectory
csv_path	String	Path to recorded data
time_step	Real	Time step for simulations with only_final_state = True.
simulation_time	Real	Simulation time for simulations with only_final_state = True.
parameters	List<RD>	Parameters to optimize for
input_variables	List<RD>	Variables to read from csv file, in same order as in csv
result_variables	List<RD>	Not required. Reference position for only_final_state = True
tracked_objects	List<Scene.Node>	Nodes/Bodies for which the error is computed
kinematic_bodies	List<RigidBody>	Not required. Bodies to animate using recording

Table 1: Required input for using *ClickOptimize*. The abbreviation RD stands for Random Dimension within the *Physics.ExperimentUniformRandomDimension* BRICK module.

various order sequences. Aside from the initialization of the controller, this FSM consists of a single state called **Operation**, which internally contains the following methods:

- **configure**: used to configure the state, it is only called once, and is used to load the successive configuration;
- **start**: this is used to initialize the controller, it is only called once;
- **run**: this is the main function and is called once per iteration loop until the state is over;
- **teardown**: this is a cleanup function called after run, and completes the controller

As the BOX scenario does not deliver an automated motion planner, packing algorithms, or vision algorithms, the motions are programmed manually, by use of the AGX Dynamics simulation environment. The details of this procedure will be discussed in Section 4.2, and will result in a set of *waypoints*, each expressing the position and orientation of the control point of the robot, potentially including the control of various inputs and outputs. Given the robot model, the mc_rtc framework allows control of the robot to reach these waypoints by means of so-called *end-effector tasks*. However, without specifying the motion between different waypoints, the robot may follow an undesired trajectory, as a result of an optimization executed by the QP-controller. To prescribe the motion of the robot between these waypoints, it is required to specify the path to follow between two consecutive waypoints and specify a time law. The implemented trajectory planner is discussed in the next section.

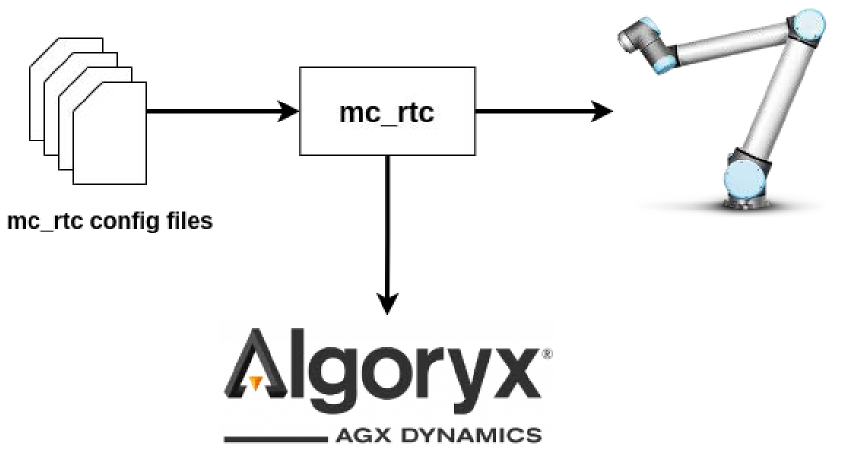


Figure 4: Schematic view of the programming process of the robot waypoints and the relationship between motion generation components and the robot and simulation environment. One mc_rtc configuration file is generated per sequence, containing the joint positions and desired IO states for each waypoint.

2.3.1 Trajectory planning

The optimization problem solved by the QP-controller in mc_rtc happens at the joint level. Therefore, simply giving a Cartesian space set-point to the controller might result in the execution of undesired motions in Cartesian space, especially when high rotations are involved. Hence, as explained in Section 2.3, one has to specify the motion between two waypoints.

The simplest and fastest way to move from two points A and B in Cartesian space is in a straight line. We have, therefore, implemented a trajectory generation module that creates a straight line path between two consecutive waypoints and rotates the tool control point (TCP) about a fixed axis $\mathbf{u}$ an amount θ radians. Then, a time law is assigned to the path such that the generated motion reference is the fastest possible while satisfying some dynamic constraints, e.g., maximum linear and angular velocity and acceleration of the TCP. The translational and rotational motions are computed separately, guaranteeing they satisfy the imposed velocity and acceleration constraints. Then, the time duration of both motions are compared and the longest (slower motion) is kept, while the fastest is then recomputed to last as much as the other one; in this way both motions will terminate at the same time, i.e., the rotational motion finishes at the moment the TCP reaches the waypoint's position. Finally, the controller is fed with the obtained continuous time trajectory sampled at any control step.

More in detail, given two waypoints in Cartesian space ${}^W\mathbf{H}_A = ({}^W\mathbf{R}_A, {}^W\mathbf{o}_A), {}^W\mathbf{H}_B = ({}^W\mathbf{R}_B, {}^W\mathbf{o}_B)$ defined as homogeneous matrices expressed in the inertial world frame (robot's base frame), one can define linear and angular displacement functions parameterized by displacement scalars $s_o, s_\theta \in [0;1]$ as

$$ {}^W\mathbf{o}(s) = {}^W\mathbf{o}_A + s_o({}^W\mathbf{o}_B - {}^W\mathbf{o}_A) \tag{4} $$
$$ {}^A\boldsymbol{\varphi}(s) = s_\theta \theta \mathbf{u}\,, \tag{5} $$

where $\theta\mathbf{u} = \log\left({}^W\mathbf{R}_A^{-1}\,{}^W\mathbf{R}_B\right)^\vee$ is the rotation vector representation of the rotation that brings frame A to coincide with frame B, meaning that ${}^W\mathbf{R}(0) = {}^W\mathbf{R}_A$ and ${}^W\mathbf{R}(1) = {}^W\mathbf{R}_B$. The translational and rotational arc length displacements are defined respectively as $L_o = \|{}^W\mathbf{o}_B - {}^W\mathbf{o}_A\|$ and $L_\theta = \theta$, while the unit vector of directional cosine of the line is given by

$$\bar{\mathbf{u}} = \frac{{}^W\mathbf{o}_B - {}^W\mathbf{o}_A}{\|{}^W\mathbf{o}_B - {}^W\mathbf{o}_A\|} = \frac{{}^W\mathbf{o}_B - {}^W\mathbf{o}_A}{L_o}. \tag{6}$$

At this point, it is possible to define a time parameterization for the translational and rotational motion as $s_o = \sigma_o(t)/L_o$ and $s_\theta = \sigma_\theta(t)/L_\theta$ respectively, with $\sigma_o \in [0; L_o], \sigma_\theta \in [0; L_\theta]$. The rotational motion in rotation matrix form expressed with respect to the inertial frame can then be computed as

$${}^W\mathbf{R}(s) = {}^W\mathbf{R}_A\,\mathbf{R}(s_\theta\theta\mathbf{u}) = {}^W\mathbf{R}_A\,\mathbf{R}(\sigma_\theta(t)\theta\mathbf{u}/L_\theta) = {}^W\mathbf{R}_A\,\mathbf{R}(\sigma_\theta(t)\mathbf{u}) \tag{7}$$

and equivalently the translation vector as

$${}^W\mathbf{o}(s) = {}^W\mathbf{o}_A + \sigma_o(t)({}^W\mathbf{o}_B - {}^W\mathbf{o}_A)/L_o = {}^W\mathbf{o}_A + \sigma_o(t)\bar{\mathbf{u}}. \tag{8}$$

The time laws are defined following the well-known *bang-coast-bang* acceleration profile which leads to a trapezoidal velocity profile and a linear segment displacement with parabolic blends and have the following expressions

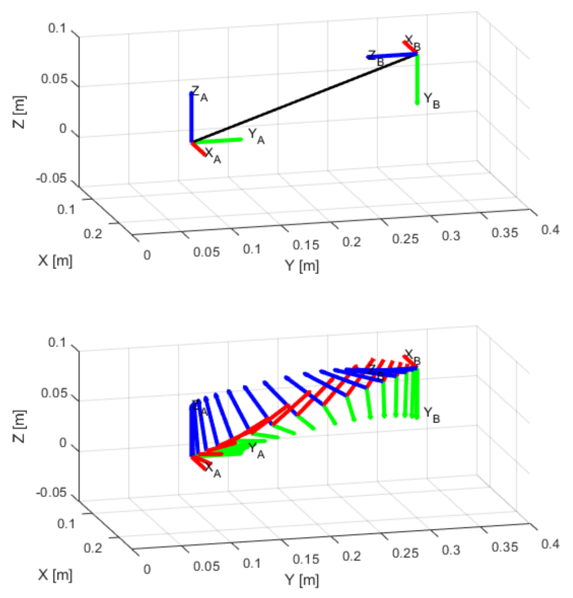
$$\sigma_o(t) = \begin{cases} \frac{1}{2}a_{max}\,t^2 & 0 \le t < t_s \\ v_{max}\,t - \frac{1}{2}\frac{v_{max}^2}{a_{max}} & t_s \le t < T - t_s \\ L_p - \frac{1}{2}a_{max}(T-t)^2 & T - t_s \le t \end{cases} \tag{9}$$

$$\dot{\sigma}_o(t) = \begin{cases} a_{max}\,t & 0 \le t < t_s \\ v_{max} & t_s \le t < T - t_s \\ v_{max} - a_{max}(t - (T - t_s))^2 & T - t_s \le t \end{cases} \tag{10}$$

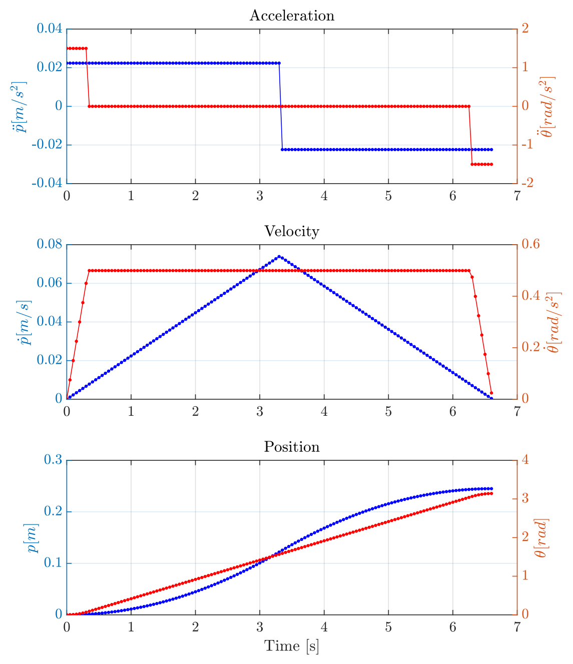
$$\ddot{\sigma}_o(t) = \begin{cases} a_{max} & 0 \le t < t_s \\ 0 & t_s \le t < T - t_s \\ -a_{max} & T - t_s \le t \end{cases} \tag{11}$$

analogous expressions hold for $\sigma_\theta(t), \dot{\sigma}_\theta(t)$, and $\ddot{\sigma}_\theta(t)$. An example of a linear trajectory in Cartesian space that is aligned in duration for the translational and rotational part of the motion while satisfying the imposed dynamic constraints is shown in Figure 5a while the corresponding displacement, velocity and acceleration profiles are plotted in Figure 5b. Adaptations to the previous set of equations were made to account for corner cases.

More sophisticated trajectories can be generated using the already existing waypoints, blending them in a unique motion and speeding up the entire pick&place process. Since no trajectory planner is integrated in the current pipeline, we implemented the simplest planner possible which is comparable with what an operator can do using the robot's teach pendant. In this report, we did not focus on more complex trajectory planning, as this is out of scope for the BOX scenario.



(a)



(b)

Figure 5: Example of a linear Cartesian trajectory using Linear Segments with Parabolic Blends (LSPB) for minimum time trajectory generation. (a) on the top figure the initial and final TCP poses are shown as well as the linear segment connecting them. At the bottom, a sequence of rotations steps along the linear segment are drawn. (b) Acceleration, velocity and displacement profile for both the translational and rotational displacement. In the example, the translational part of the motion (blue) is generated in two phases (acceleration-deceleration) while the rotational part (red) is generated in three phases (acceleration-constant speed-deceleration).

3 Scenario Specification and Experimental Setup

This section elaborates on an industry-relevant scenario devised based on the premises of Section 1, which is used to test and validate the I.AM.-developed components presented in Section 2.

3.1 Scenario Specification

In order to validate the I.AM. components developed for the BOX scenario (Section 2) against a real-world application, a representative industrial use-case was devised, inspired by the current industry standard (discussed in Section 1.3) and the experience that both Smart Robotics and Vanderlande have in the automated Item Picking industry.

The scenario consists of the fulfillment of orders in a warehouse by a robotic arm. The robot should pick items from a source container and place them in a target container. In the context of the BOX scenario, the items' properties are based on real industry data (Section 3.1.1) and the order in which the items are fed to the robot is specified by an *item sequence* (Section 3.1.2). In this scenario, five item sequences will be evaluated.

The experimental setup used to validate the developed I.AM. components against this scenario is elaborated in Section 3.2. The implementation and testing methodology of this validation scenario with BOX will be explored in detail in Section 4.

This scenario will be also implemented and tested by a human operator and on a state-of-the-art robotic system (Smart Robotics Item Picker), to serve as a benchmark against the performance of the BOX experimental setup. This can be found in Section 5.

3.1.1 Item set generation

In order to evaluate the performance of boxing in realistic and industry-relevant settings, the item set and packing orders used in these test scenarios are based on real-world data of items packed by robots at a customer site, provided by both Smart Robotics and Vanderlande. To protect the privacy of the customers, this data has been anonymized beforehand - only item sizes and masses were provided.

This data proved to be very extensive, with an immense amount of different items - each with their own size and mass values. Using the data in this form as direct input to build an item set is not feasible. For this reason, an effort was made to condense the raw data items into a much smaller set of representative items. This is possible due to the fact that several items have very similar sizes and masses, and can be grouped into one representative item.[7] After performing this reduction, the dataset was still quite large with around 200 representative items. Therefore, another reduction step was devised, in which only the most frequent representative n items were to be used. In an effort to keep the item set with a manageable size, n was chosen to be 4. Afterwards, it was verified that the subset of the 4 most frequent representative items could approximate the sizes of most of the items of the raw dataset within 20% of their size. This was deemed acceptable.

To keep the item set relevant to the scope of the project, their shapes are considered to be rectangular

[7]To illustrate this, one may consider the example of a mobile cellphone, where although several brands and models exist (and each one with their own packaging), their packaging sizes and masses are quite similar to each other and can be represented by an average size and mass.

prisms - which can be realized as cardboard boxes. The 4 representative items are shown in Figure 6. The nominal properties of these representative items can be found in Appendix A, Table 8.

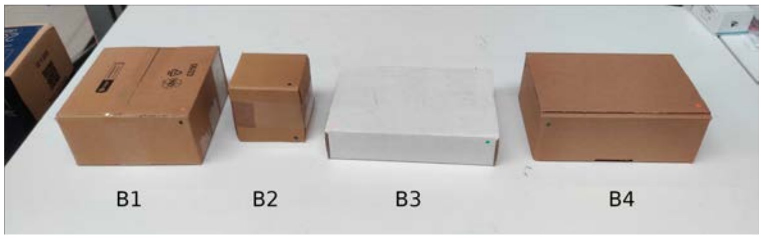


Figure 6: The 4 base items types.

3.1.2 Item sequences

An item sequence is defined as the order in which items are provided to the item-placing agent, to be placed inside the container. It simulates a real-world scenario in which an external agent instructs the robot to pick and place a set of items in a particular order, such as by an automated warehouse management system. The item sequence is independent of where the robot should place the item in the target container. It simply defines the order in which the items are to be fed to the robot, and the robot is generally free to choose how to handle the item. In the specific case of the BOX scenario tests, the item sequences will be used as a means to standardize the inputs across the different test scenarios presented in Section 5.1 such that their performances can be compared with each other.

Using the 4 base items discussed in Section 3.1.1, 5 item sequences were manually generated to approximate real-world scenarios where multiple instances of a particular item are requested in succession, or only some of the items are requested initially and the others are added at the end of the sequence. These sequences have the minimum number of boxes required to reach a theoretical filling degree of 90%. This is enough since, for the generated sequences and dimensions of the items and the container, the optimal filling degree should be no higher than 90% (which is also in line with the findings reported in Deliverable D7.3 [1]). As a result, unnecessary work constructing superfluous boxes is avoided.

Item sequences are generated based on item type (B1, B2, etc.) in order to allow reuse of items between sequences in a way that minimizes the number of item specimens manufactured in the lab. A total of 18 individual items were created in order to accommodate all generated sequences: 4 of type B1, 6 of type B2, 4 of type B3 and 4 of type B4.

Each item is assigned a unique ID as well, as they will not have exactly the same properties due to manufacturing errors (even if they are of the same type). The unique ID of the items follows the TU/e Robotics Lab convention of naming test boxes as **BOXxxx**, where **xxx** is a unique number. BOX013 to BOX016 are boxes of type B1, BOX017 to BOX022 are of type B2, BOX023 to BOX026 are of type B3, and BOX027 to BOX030 are boxes of type B4 (please consult Table 8 for the nominal properties of each box type).

The sequences used for testing are presented in Table 11.

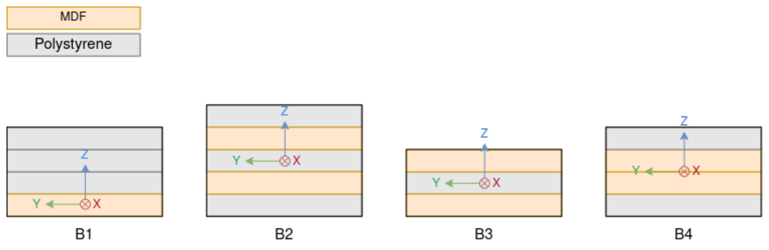


Figure 7: A schematic representation of the internal structure of each of the 4 box types, along with the location of their respective centers of mass, as well as their frames of reference.

3.1.3 Item manufacturing

Taking into account the item sequences generated in Section 3.1.2 (see Table 11), a total of 18 test items were manufactured: 4 of type B1, 6 of type B2, 4 of type B3 and 4 of type B4. These items consist of cardboard boxes filled with layers of MDF laminated wood ($\rho = 620kg/m^3$) and polystyrene ($\rho = 14kg/m^3$), which are assumed to be of uniform density. The cardboard boxes themselves are also assumed to have uniform density, and air is negligible. The MDF layers were used to ensure the boxes have the desired weight, while the polystyrene layers were used as a filler - to keep the MDF layers from moving around.

The internal layer composition of each box, as well as their respective center of mass frames of reference, are represented in Figure 7.

Constructing the boxes in this layered fashion also allows us to take advantage of symmetries when calculating the moments of inertia I of each box. These can be calculated as in Equation 14:

$$m_k = x_k \cdot y_k \cdot z_k \cdot \rho_k \tag{12}$$

$$I_k = \frac{m_k}{12} \begin{bmatrix} y_k^2 + z_k^2 & 0 & 0 \\ 0 & x_k^2 + z_k^2 & 0 \\ 0 & 0 & x_k^2 + y_k^2 \end{bmatrix} + m_k d_k \mathbf{I}_{3\times3} \tag{13}$$

$$I = I_{box,empty} + \sum_{k=1}^{N} I_k \tag{14}$$

where for a given box material layer k (composed of either MDF or polystyrene), (x_k, y_k, z_k) are the layer dimensions, ρ_k is the density of the layer, m_k is the mass of the layer, d_k is a scalar distance from the center of mass of the layer to the center of mass of the filled box, and I_k is the layer's tensor of inertia (represented in the center of mass frame of the filled box). $\mathbf{I}_{3\times3}$ is a 3×3 identity matrix. $I_{box,empty}$ is the inertial matrix of only the outer cardboard shell of the box. N is the total number of layers in a box.

The properties of the 18 fabricated items used during testing are compiled in Table 9. The moments of inertia I were computed for each item in the item set, and are shown in Table 10.

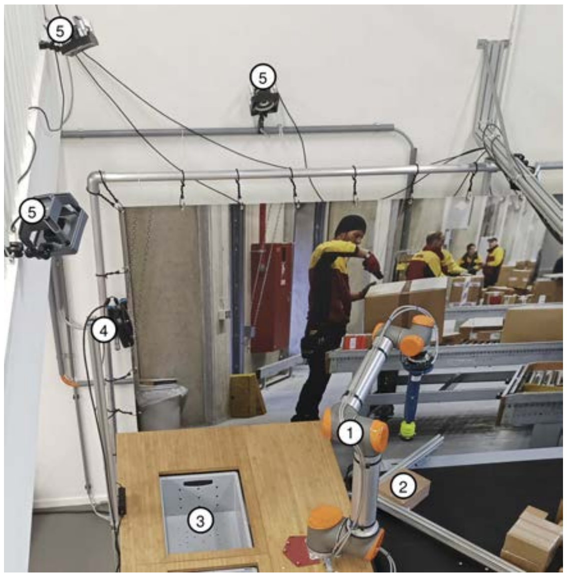


Figure 8: Overview of the test setup at the Vanderlande robotics lab, used for the Robotic packing with I.AM. test scenario. **1:** UR10 robot; **2:** Box pick location; **3:** Box place location (container); **4:** Zivid 2 3D vision camera; **5:** OptiTrack cameras.

3.2 Experimental Setup

The setup at Vanderlande Robotics Lab located at the Meulensteen House of Robotics on the Eindhoven University of Technology campus is used for validating the I.AM.-developed components (Section 2) in the experimental scenario presented in Section 3. This setup closely relates to similar systems being deployed by both Smart Robotics and Vanderlande. The setup is equipped with the following components:

1 A Universal Robots UR10 cobot, used as the manipulation agent that will interact with the boxes during packing. It is equipped with a BOTA SenseONE EtherCat force torque sensor. Under it, a Smart Robotics GS002 vacuum gripper, with a Piab piGRIP 70mm diameter suction cup, with a soft foam lip. The robot will be controlled using mc_rtc.

2 A conveyor with a box-alignment tool, which is a metal frame used to position the boxes on the conveyor for pick-up.

3 A 570x370x250 mm container, where the items will be placed. This container is integrated in a custom-made container port.

4 A Zivid 2 3D vision camera, looking directly into the target container, captures the state of the container after each place. It provides both RGB images and point clouds of the scene, that can be used for posterior analysis. The camera is manually triggered after each box placement, and as soon as the robot moves out of view.

5 An OptiTrack motion capture system setup consisting of 6 cameras, tracking the pose of each box, the gripper, the suction cup, and the container through time. Each box, the gripper and the suction cup are equipped with a unique pattern of OptiTrack markers that allows for unambiguous tracking of each component. However, when the boxes are placed inside of the container, tracking becomes difficult due to the reduced visibility of the markers by the OptiTrack cameras. The Zivid

2 camera provides help in this situations. Point cloud data can be later used to track the pose of the boxes inside the container.

These components are also indicated in Figure 8. The UR10 cobot and the gripper attachment are selected for the BOX demonstrator for direct comparison with the state of the art at Smart Robotics, since Smart Robotics's item picking application uses the same components. The container port and conveyor are similar to those seen in industry, as discussed in Section 1.3. Both are the same as those used in the I.AM. TOSS scenario [6]. The pick location is not another container, as one might expect; it is an isolated corner in the conveyor. This makes it easier to feed the boxes one-by-one to the robot in a controlled way, and allows for the boxes to be positioned always at the same position, against the reference corner. This is important as a vision algorithm deciding where to pick the boxes is out of scope for this project (and the BOX scenario only focuses on the placing of the items anyway). We want a robust way of picking the items consistently, to ensure that positioning errors at the pick pose do not propagate into the place pose.

Several systems are used in conjunction to collect real-world data during robot motions, for later comparison with AGX Dynamics. The force-torque sensor is mounted between the gripper and the UR10 flange to allow for those measurements during motions. The Optitrack system is used to get the positions of all components relative to a common world frame. The Zivid 2 camera is positioned to look into the target container from above in order to assess the current state of the contents of the target container.

This container is representative of the type and size of containers found in warehousing systems where pick-and-place robotic solutions operate. Its inner dimensions are 570x370x250 mm, and it is shown in Figure 9.

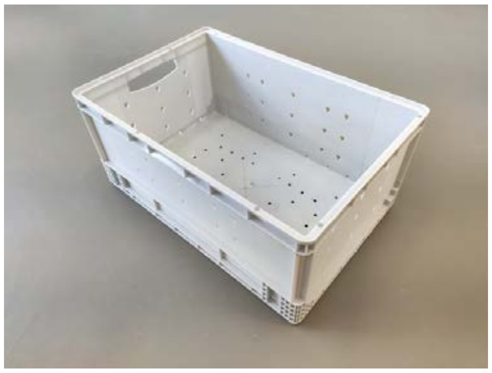
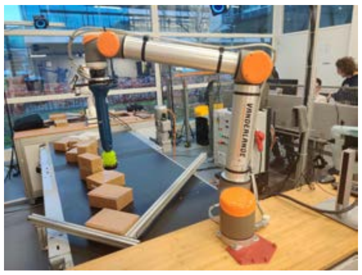


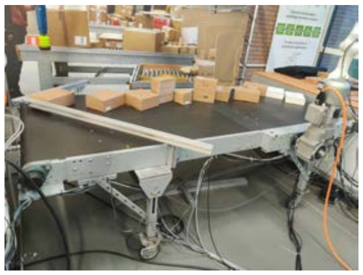
Figure 9: The container where items were placed in all test scenarios.

An overview of the individual main components of the test setup is shown in Figure 10.

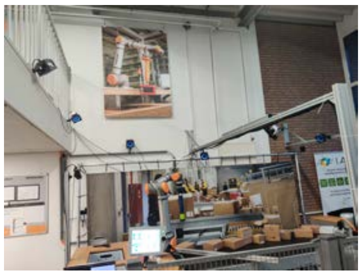


(a)

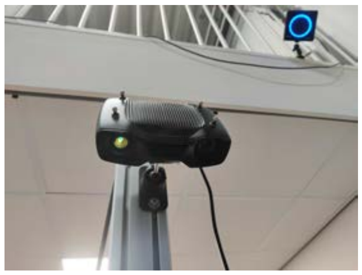
(b)



(c)



(d)



(e)

Figure 10: Pictures of the BOX scenario robot architecture components. The UR10 robot (a), the container port with target container on the top (b), the conveyor with item sequence (c), the OptiTrack system (d), and the Zivid 2 camera (center) positioned above the target container and an Optitrack camera (upper-right) (e).

4 Experimental validation and comparison

In this section, we will describe the experimental validation of the developed framework as discussed in Section 3. First, in Section 4.1 we will describe the simulation environment, specifically tailored to match the real-world setup. Then, in Section 4.2, we will describe how we generated the motions of the robot in the simulation environment for the different sequences in order to pack the boxes inside the container. In Section 4.3, we discuss the transition from simulation to the real setup and explain how we collect the data from experiments. Finally, in Section 4.4, we take the collected experimental data and compare it to the simulation results for some specified key moments.

4.1 Simulation Environment

The simulation environment closely replicates the real setup presented in Section 3.2. In this section, we will describe the various components that compose the setup and how they are modeled in the simulation environment.

4.1.1 Robot description model

The robot used in the experiments, a Universal Robots UR10, is described by means of a URDF file. This XML-based file contains all the information about the kinematic chain composing the robot, as well as the inertial parameters of the different links and components. The official robot description model gives an ideal description of the robot accordingly to its nominal Denavit-Hartenberg (DH) parameters provided by the robot manufacturer. However, in reality the robot is constructed with certain manufacturing inaccuracies due to assembly processes and part tolerances, resulting in the need for a custom URDF file that accounts for the kinematic calibration of the specific instance of the robot at hand (the results of a calibration procedure performed by the manufacturer can be downloaded from the robot's controller[8]).

The robot's calibration file contains the deviations from the nominal DH (δDH, a matrix $\in \mathbb{R}^{4*6}$) parameters of the robot obtained from an optimization procedure that minimizes the error between the actual end-effector's pose ${}^{A}\mathbf{H}_{EE}$ (measured with very accurate measurement devices at the manufacture site) and the forward kinematics function $\mathcal{K}(DH + \delta DH, q)$ for a given set of joint configurations (q_k):

$$\delta DH^* = \arg\min \left\| \sum_{k=1}^{N} ({}^{A}\mathbf{H}_{EE_k})^{-1} \mathcal{K}(DH + \delta DH, q_k) \right\|^2 \tag{15}$$

This procedure only accounts for the accuracy of the end-effector's pose without caring about the placement of the intermediate frames along the kinematic chain. As a result, the optimal solution (δDH^*) may have very large values - as it is in our case. This is not a problem for computing both forwards and inverse kinematics (indeed the robot internally uses those values), but becomes an issue when one wants to build a simulation model of the robot. The location of the i-th joint as well as the i-th link's mesh of the visuals, collisions, center of mass and inertia, are all expressed with respect to the joint frame $i - 1$, and if there are large values in the δDH, they will lie quite far from where they should be. To overcome this issue, we have performed a penalty-based optimization to obtain the smallest set of DH parameter

[8] See the *www.universal-robots.com* website on robot calibration.

deviations ($\overline{\delta DH}$) that still satisfy (15):

$$\overline{\delta DH}^* = \arg\min \left\| \sum_{k=1}^{N} (\mathcal{K}(DH + \delta DH, q_k)^{-1} \mathcal{K}(DH + \overline{\delta DH}, q_k) + \overline{\delta DH} \right\|^2 \tag{16}$$

Once we obtained the new set of calibration parameters via the penalty-based optimization, we built a new custom URDF of the robot, whose Cartesian poses are now coincident with the real robot for any given joint configuration.

At the flange of the robot, a 3D printed vacuum gripper designed by Smart Robotics is mounted right after a 6D force/torque sensor. Due to thermal warping during the 3D printing process, the gripper bends in the longitudinal direction of printing, causing a deviation of the position and orientation of the tool tip with respect to the CAD. By mounting a conic end-effector on the robot's flange, we used it as a measurement device and performed a calibration to compensate, in the simulation, for this offset.

Definitions in BRICK: BRICK uses the same source of truth, that same UR10 URDF file, as the controller for the robot definition. The BRICK robot model has been extended with a 3D force and torque sensor and the implementation of the flexible suction cup.

4.1.2 Environment description

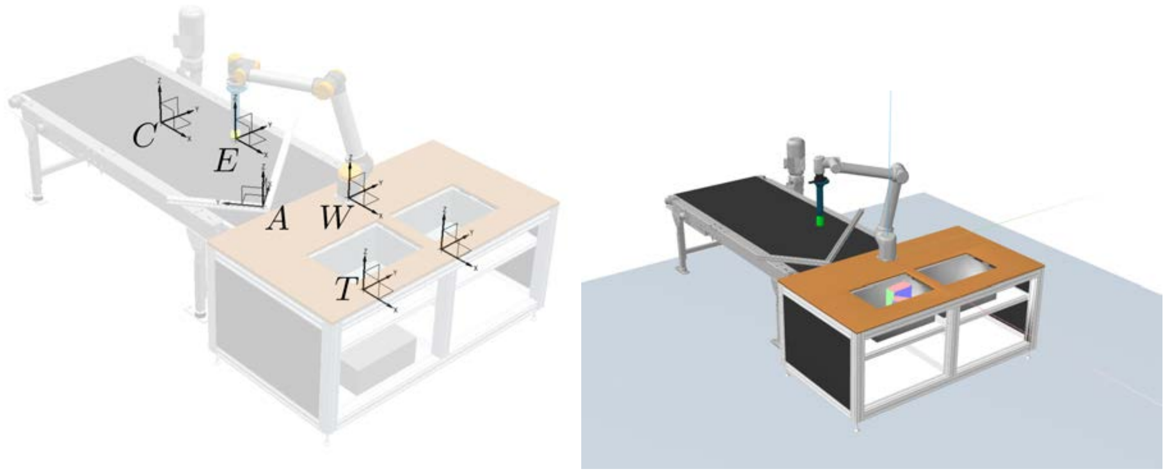
In Figure 11a, the simulated setup is schematically depicted, including the different reference frames. Frame A defines the alignment tool, which indirectly defines the pick-up position and orientation of the boxes. Frame E is the frame that defines the control point of the robotic arm. Frame W is located at the origin of the robot and also used as world frame. Frame C defines the conveyor, and frame T defines the container. Given the world frame W, we can express the poses of the alignment tool, the container, and the conveyor with transformation matrices given as

$${}^W\mathbf{H}_A = \begin{bmatrix} {}^W\mathbf{R}_A & {}^W\mathbf{o}_A \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, {}^W\mathbf{H}_T = \begin{bmatrix} {}^W\mathbf{R}_T & {}^W\mathbf{o}_T \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, {}^W\mathbf{H}_C = \begin{bmatrix} {}^W\mathbf{R}_C & {}^W\mathbf{o}_C \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \tag{17}$$

with ${}^W\mathbf{o}_A = \begin{bmatrix} -0.353 & -0.418 & -0.0355 \end{bmatrix}^T$, ${}^W\mathbf{o}_T = \begin{bmatrix} 0.505 & -0.350 & -0.282 \end{bmatrix}^T$, and ${}^W\mathbf{o}_C = \begin{bmatrix} -1.76 & -0.16 & -0.105 \end{bmatrix}^T$ expressing the positions, and

$${}^W\mathbf{R}_A = \begin{bmatrix} -0.6428 & -0.7660 & 0 \\ 0.7660 & -0.6428 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{18}$$

and ${}^W\mathbf{R}_T = {}^W\mathbf{R}_C = \mathbf{I}_{3\times3}$ expressing the orientation. Note that the rotation expressed by (18) represents a pure rotation around the z-axis of 2.269 radians. These poses are obtained from manual measurements in the lab. Such measurements were performed utilizing the UR10 robot, equipped with a conic end-effector of known dimensions, as a measurements tool. The various boxes, as discussed in Section 3, are modeled as fully rigid boxes, and their mass and inertia properties are copied to the simulation environment. In the simulation, they arrive via the conveyor belt to the box alignment tool, according to the sequences defined in Section 3.



(a) (b)

Figure 11: The modeled setup. Picture of the modeled scene indicating the different frames (a) and picture of the modeled scene in AGX Dynamics (b).

Contact Parameters			
Object 1	**Object 2**	**Friction μ**	**Restitution e_N**
Box	Box	0.2	0.05
Box	Container	0.24	0.1
Box	Workbench	0.15	0.1

Table 2: Contact parameters as defined in the BRICK model

4.1.3 Contact description

To properly model the contact interactions, we have to further specify the contact parameters. More specifically, we assign a material to the box and each element that the box can interact with, and we define a coefficient of friction and coefficient of restitution that describes that contact. In Table 2 these contact parameters are described. The friction between the box and the container is estimated from a set of various experiments as described in [9]. The other parameters are obtained from simple tests from which the parameters are estimated. The default friction model in AGX Dynamics is the Box-Friction[9], which is known to cause inaccuracies when the direction of the friction force is not along one of the principle axis of the box-friction model [9]. Instead, as computation time is not a hard constraint, we use the more realistic, however, computationally heavier, Iterative Projected Cone Friction model[10].

4.2 Motion Primitives and Simulation Data

Given that a suitable vision system, item packing algorithm, and motion planner are considered out of scope for this project (as these can be integrated at a later stage), the motion primitives for the items are

[9]See the *www.algoryx.se* website on Box Friction.
[10]See the *www.algoryx.se* website on Iterative Projected Cone Friction.

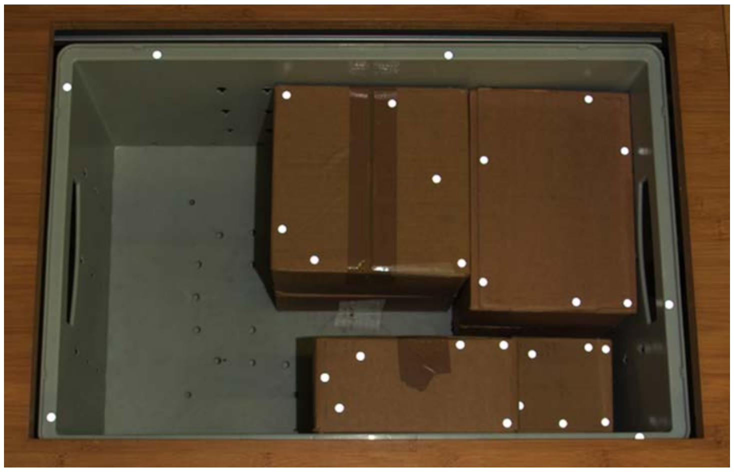


Figure 12: Container status after the placement of 6 boxes. The 7th box (Box016) can not be fit in the “bottom-left” corner (the largest free area in the container) in the orientation it arrives at the infeed conveyor. Therefore, it is flipped and fitted in such a corner.

to be chosen and programmed by a human operator using the simulation environment. The operator will, however, try to mimic the behavior of a fictitious item packing algorithm. This fictitious packing algorithm should follow some guidelines:

- It cannot look-ahead, i.e., there is no knowledge of the items that are to come. It only has knowledge of the item currently being processed and the items already placed in the container.
- Placement is permanent: once an item is placed in the container, it cannot be re-picked and placed somewhere else.
- It has knowledge of and can exploit the potential benefits of I.AM.-developed technologies. For example, it may choose to place items in tight spaces and use the compliance and modeling of the suction cup to its advantage, pushing the held item against other items already placed in the container.
- It may flip items around before placing them, by ordering a flip-repick-place sequence of motions. Flipping items around in a consistent and predictable way is enabled by I.AM.-developed technologies, namely the modeling of the suction cup. Figure 13 illustrates the sequence of motions that make the flipping of items possible.
- It will not place items in a potentially unstable manner inside the container. For example, if an item is so thin on one of its sides as to significantly increase the risk of tumbling over when placed on that side, the algorithm will prefer to flip the item around such that the item is placed on a more stable side.

The defined sequences of boxes as discussed in Chapter 3 are copied towards the simulation environment. According to the sequence, the boxes arrive on the conveyor and end up at the box-alignment tool located on the conveyor (frame A in Figure 11a), allowing the robot to pick them up. The waypoints that the robot should follow to pick and place each item are then defined in accordance with the fictitious packing algorithm as described above.

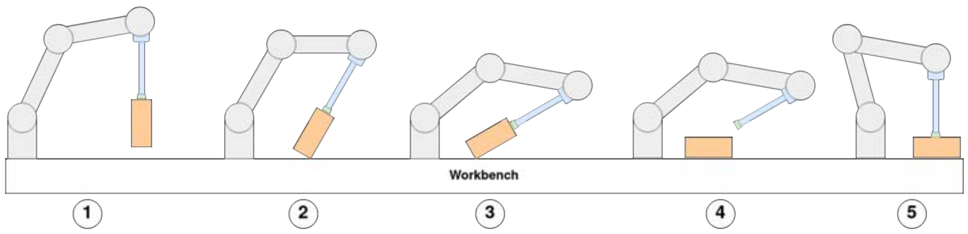


Figure 13: Sequential representation of an item flip motion, as performed by the robot. **1:** Robot approaches the workbench holding an item; **2:** Robot begins to rotate the box around the pivot point defined by the edge of the item contacting with the workbench; **3:** Robot forces the item to almost rotate 90°, by sliding the item on the workbench; **4:** Robot releases the item, and the item comes to rest in a new orientation; **5:** Robot re-picks the item in this new orientation.

In certain cases, the choice is made to change the orientation of the box with which it arrives on the conveyor. This is typically done when, due to a change in orientation, the box can be placed at a more convenient place inside the container. Figure 12 shows a situation where this is the case. The container already contains 6 boxes. The box that arrives next is of similar dimensions to the previously placed box. However, the orientation in which it arrives on the conveyor does not allows for placement inside the container. Therefore, the decision to flip the box is made, in order to change its orientation. The procedure of flipping an arbitrary box is shown in Figure 13.

The design of the motion primitives results in a set of waypoints, expressing the Cartesian position and orientation of the control point of the robot arm (frame E in Figure 11a). At each waypoint, a decision can be made to control certain I/O, which in this case includes only the vacuum of the suction cup, allowing the arm to pick up and release objects. For each sequence, the motions primitives and I/O control inputs are stored in dedicated CSV files, allowing the controller to easily switch between different sequences.

4.3 Experimental data collection

Having obtained the motion primitives from Section 4.2, we can now execute the controller on the real setup, described in Section 3.2. In order to run the controller on the real setup, the following procedure is executed for each item sequence (see Figure 14 for flowchart):

1 The mc_rtc control application is started.

2 The next item in the sequence is placed against the box aligner.

3 The robot picks the item from the pick-up point and places it in the target container according to the predefined motions as obtained from simulation.

4 An operator manually takes a depth image with the Zivid 2 camera to capture the state of the container.

5 Steps 2 till 4 are repeated until no more items can be placed or there are no more items listed in the sequence.

6 The collected data from the robot's operation, the depth and RGB images of the target container, and OptiTrack data, are stored for later analysis.

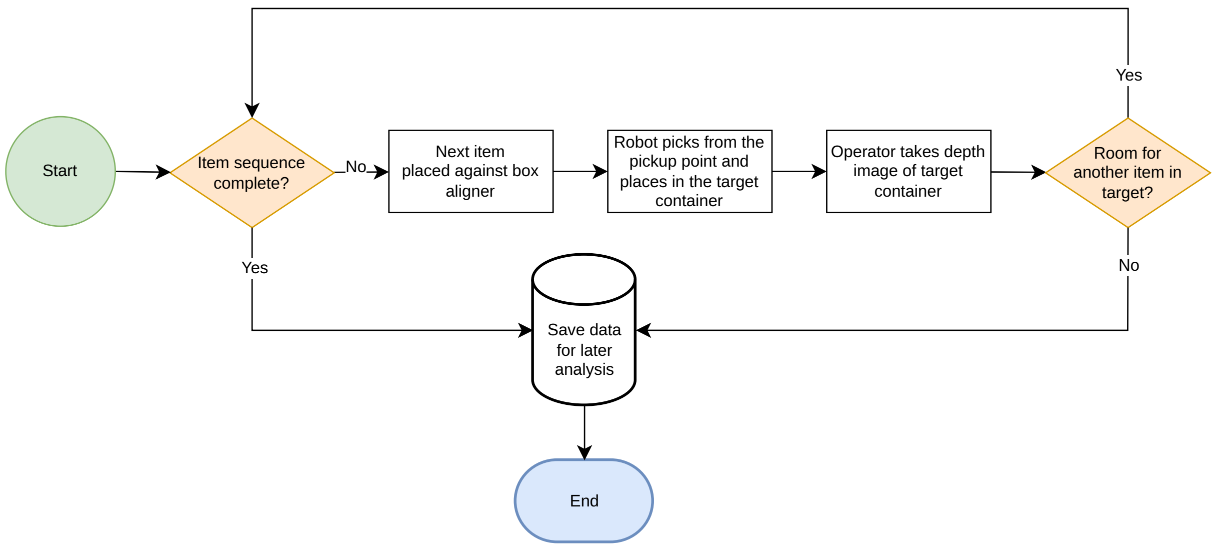

Start
Item sequence complete?
No
Next item placed against box aligner
Robot picks from the pickup point and places in the target container
Operator takes depth image of target container
Room for another item in target?
Yes
Yes
No
Save data for later analysis
End

Figure 14: I.AM. BOX demonstrator test procedure flowchart.

Before the start of each new sequence, a calibration procedure is executed to calibrate the force/torque sensor, after which each sequence is recorded three times in order to account for any uncertainties that may occur. For each recording, we collect data from the UR10 encoder and I/O output, the Zivid, the force/torque sensor, the mc_rtc control log, the motion capture system, and a video camera used to film the experiment for reference. The raw data files are then converted into an archive, according to the structure described in Deliverable D1.4 [10] and [11].

4.4 Comparing simulation to experimental data

The I.AM. BOX demonstrator is composed of a test setup (see Section 3.2) and a model run in simulation as described in Section 2.2) and this chapter. The accuracy of the simulated model indicates whether a future motion planner may use this information to predict the dynamics of the item during placement in the container. This is needed in order to avoid error-causing collisions (i.e. the robot goes into a protective stop) which would require an operator to intervene with the robot or results in items being damaged. The higher the model accuracy, the lower the margins that have to be taken into account when planning motions.

It is worth noting that all the experiments, real and simulated, were executed open-loop, i.e., once an algorithm has determined the box placement pose and the sequence of way-point to achieve it is generated by a motion planner, the sequence is executed without using any sensor feedback, which makes the result here presented to be even more remarkable. Consider furthermore that no data was used to initialize the pick pose of the boxes; only a proper scene modeling, as described in Section 4 was performed.

4.4.1 Validation Criteria

In order to compare the experimental and simulation data, we define certain performance criteria with the specific focus on the developed components and objectives. The performance criteria are listed as follows:

1 **Execution**: The generated motion primitives should be able to be executed on the real setup and, as in simulation, each sequence should be fully executable.

2 **Final Box poses**: The simulated positions and orientation of the boxes after the full sequence should match those in experiments, indicating that the final result of the filled container can be predicted with sufficient accuracy.

3 **Holding Box poses**: The simulated positions and orientations of the boxes during the holding phase should match those in the experiments, indicating the fidelity of the suction cup model during these phases.

4 **Wrench predictions**: The simulated wrenches acting on the robot should match those in the experiments, indicating the fidelity of the implementation of the robot and suction cup model during these phases.

The first criterion determines if the full sequence can be executed on the real setup as expected from the simulation. The second criterion defines to what extent the resulting container is as expected. This can be crucial in cases where the sequence can be fully executed, but boxes are misplaced and stick out of the container, which can cause problems further down the line. The third and fourth criteria focus on the fidelity of the suction cup model and the general implementation of the models into the physics engine. Such criteria evaluates the pose and wrench predictions and give an impression to what extend those predictions can be used for trajectory planning and force feedback. For these last two criteria, some key moments are selected from different sequences for which an evaluation is done. These key moments are considered as crucial for the sequence, meaning that slight deviations in the predictions can lead to failure of packing the boxes. We have listed them in Table 3.

4.4.2 Performance Validation

For each of the criteria defined in Section 4.4.1, we will evaluate the performance in the five different sequences.

Execution: The generated motions primitives were performed on the real setup and all motions were properly executed. However, due to non-modeled effects of the contact and release phase of the suction cup, a systematic error occurred for each re-pick of a flipped box. In real experiments, the box is pushed a few centimeter away by the suction cup, and this phenomena is not yet implemented in simulation. In the real experiments, this systematic error was accounted for in the controller, without changing the motion primitives. After this correction, each of the motions of each of the sequences was executed correctly. Note that in a real application, vision feedback can be used for determining the pose of the box. We recall that the use of vision is out of the scope of the project.

Final Box Poses: After running each of the sequences on the real-setup, the state of the boxes in the container was captured in order to compare to the final box's poses as obtained from simulation. In Figure 20, the final results of the boxes in the container are shown for the experiments (third column) and the simulation (forth column), for each of the 5 sequences. Furthermore, we have analyzed the error in position and orientation between the simulation and the experiments. We have listed the results in Table 4.

Box Poses: In Section 4.4.1, ten phases of interest were selected for evaluation. In this section, we will

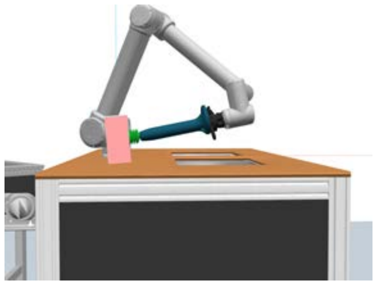
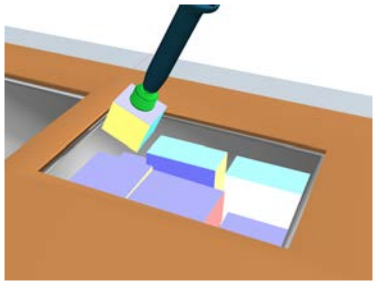
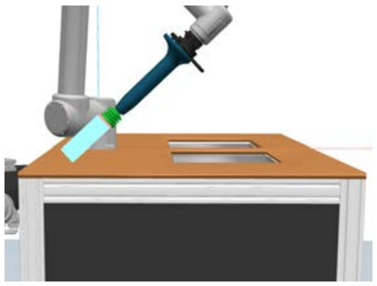
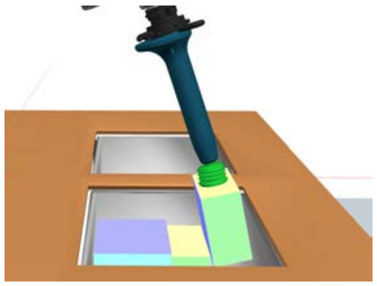
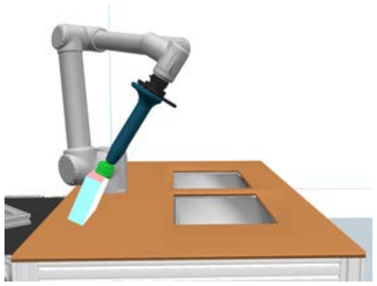
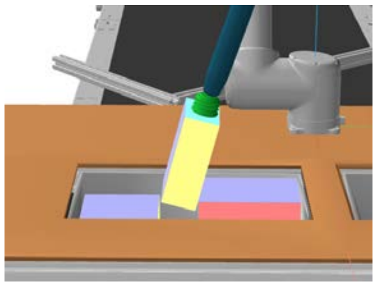
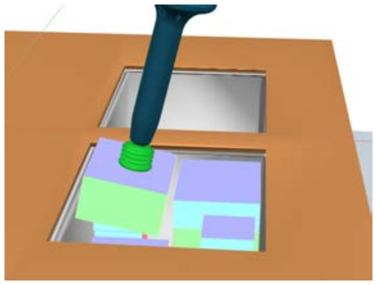
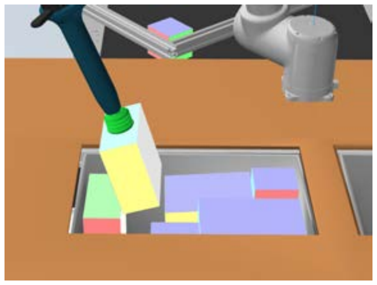
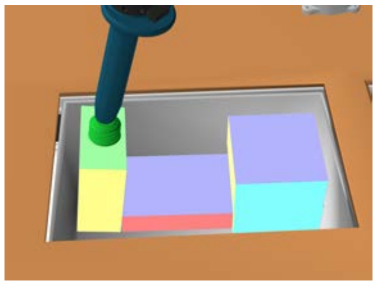
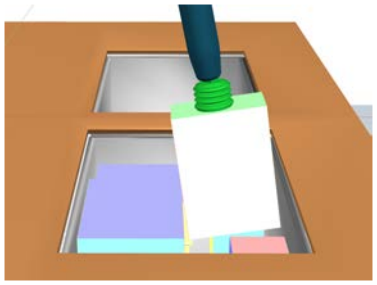
Seq.	Phase 1	Image	Phase 2	Image
1	Box016 Flipping		Box018 Packing	
2	Box026 Flipping		Box028 Packing	
3	Box025 Flipping		Box028 Packing	
4	Box029 Packing		Box015 Packing	
5	Box014 Packing		Box027 Packing	

Table 3: Selected key moments for the analysis of the pose and wrench predictions of the model. See Appendix C for the pose and wrench plots of some of the key moments in this table.

	Seq. 1		Seq. 2		Seq. 3		Seq. 4		Seq. 5	
	e_{pos} [mm]	e_{rot} [deg]	e_{pos} [mm]	e_{rot} [deg]	e_{pos} [mm]	e_{rot} [deg]	e_{pos} [mm]	e_{rot} [deg]	e_{pos} [mm]	e_{rot} [deg]
Box013	2.78	2.20	-	-	2.25	1.16	-	-	7.09	1.81
Box014	8.83	0.70	n.a.	n.a.	-	-	n.a.	n.a.	n.a.	n.a.
Box015	4.46	2.43	7.34	0.54	-	-	14.77	1.01	7.86	2.02
Box016	4.30	3.61	-	-	4.86	0.94	-	-	11.7	1.26
Box017	6.45	1.57	7.90	1.81	5.26	1.65	5.02	1.69	8.07	1.60
Box018	5.68	1.27	16.62	1.35	6.62	0.95	6.21	1.54	5.90	1.46
Box019	-	-	7.27	2.66	5.93	2.49	6.96	2.48	10.13	1.53
Box020	-	-	-	-	6.96	2.89	7.42	1.36	-	-
Box021	-	-	-	-	-	-	4.60	1.87	-	-
Box022	-	-	-	-	-	-	9.04	1.96	-	-
Box023	5.97	2.25	11.82	2.72	11.96	1.17	7.22	0.89	n.a.	n.a
Box024	8.60	n.a.	6.97	n.a.	8.88	1.18	n.a.	n.a.	14.19	n.a
Box025	-	-	8.58	2.51	8.21	3.95	11.14	1.02	16.55	1.39
Box026	-	-	10.95	1.69	14.93	1.45	16.17	1.01	8.87	1.08
Box027	6.79	1.58	n.a.	n.a.	7.21	1.85	-	-	10.48	1.64
Box028	7.67	0.78	n.a.	n.a.	n.a.	n.a.	-	-	-	-
Box029	7.91	0.84	n.a.	n.a.	-	-	16.02	5.33	-	-
Box030	9.75	1.33	-	-	7.13	2.00	6.58	2.95	-	-

Table 4: Rest-pose prediction errors for the different boxes in each sequence. The dash indicates that the specific box is not part of the sequence. When a value is n.a. (not available) means that the Mocap system has lost track of the box, preventing us from recovering the values. Note that we have collected point-cloud and RGB data (Zivid camera) of the state of the container after each box placement and that, in principle, the final pose of the object can be estimated. Due to the different levels of accuracy between the two sensors, we have chosen not to use the Zivid data in the analysis and present consistent data.

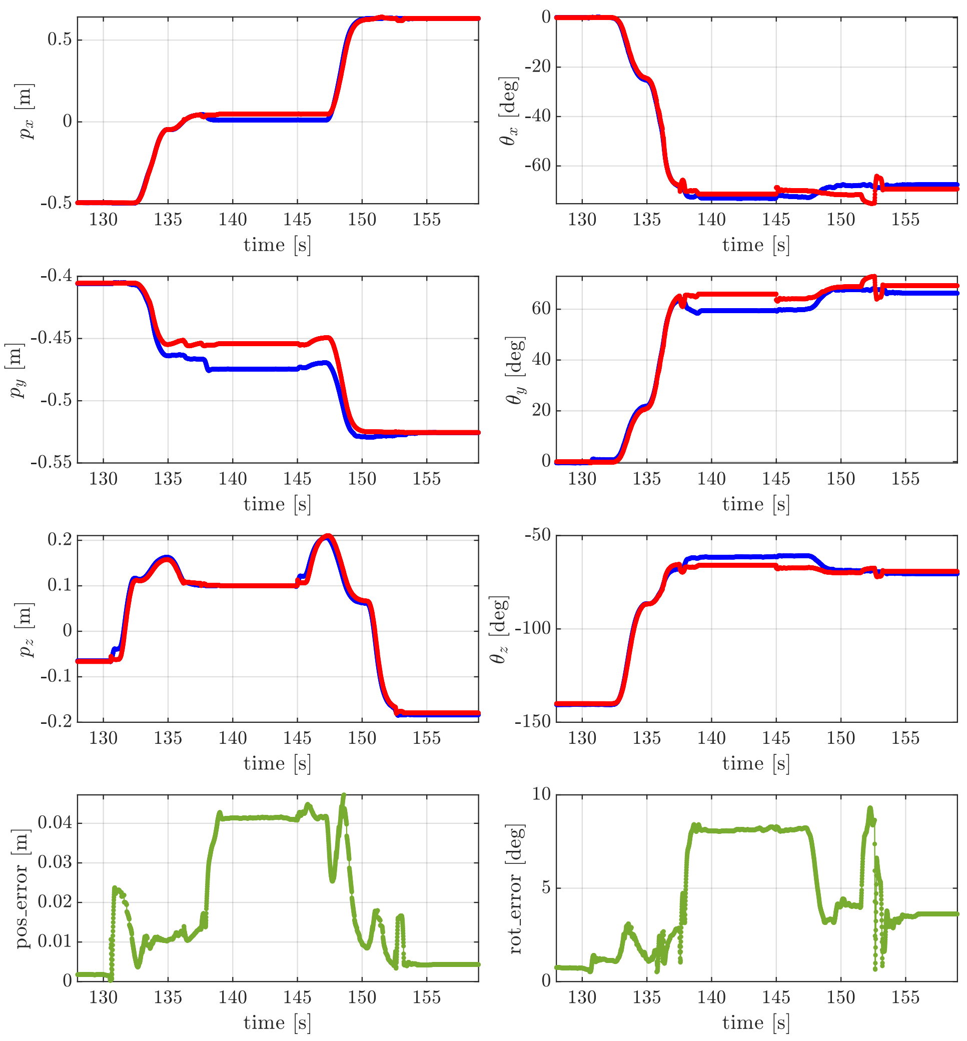


Figure 15: Resulting poses for the full motion of Box016 in Sequence 1.

specifically focus on two of those phases and analyze them in depth. The first phase of interest is flipping Box016 in Sequence 1, and the second moment of interest is packing Box027 in Sequence 5. The other 8 phases of interest are shown in Appendix C.

In Figures 15 and 16 the poses of Box016 in sequence 1 and Box027 in sequence 5 respectively, are plotted over time. On the left column, we have shown the x, y, and z-components of the position vector, and the norm of the position error for each time-step. In the right column, we have shown the three components of the rotation vector, and the norm of the rotation error over time. During this motion, there are a few moments of interest that are worth explaining. First, at $t = 129$ seconds in Figure 15, the box is being picked up and rotated while being placed on the workbench. During this motion, the position error is around 1.5cm, and the rotation error stays below 5°. This higher error during pick-up (displayed at $t = 131$ in the z-component of the position) can be explained by the fact that in the real experiment, the suction cup changes length due to the creation of vacuum the moment it is in contact with the box lifting it. This behavior is not modeled, which explained why this behavior is not shown in the simulation.

At $t = 137$, the box is being released on the workbench. In the real experiment, the vacuum is turned off, and the suction cup elongates, pushing the box approximately 2cm away. This is clearly visible in the position error, as well as in the y-component of the position vector. During this release, the orientation of the box is also changed, causing a total error of approximately $8°$. At $t = 145$, the box is re-picked. In the real experiment, this re-pick location is slightly different due to the misplacement with respect to the simulation (due to the non-modeled behavior). In the 5 seconds after, this error reduces due to the fact that the trajectories of the robot in experiments and simulation converge back to the same way-point at $t = 150$. From $t = 150$ till $t = 155$, the box is placed inside the container. It is clear that at $t = 153$, in simulation, the box slightly wobbles before coming to rest, causing a rapid change in orientation, that is not observed in the real experiment. However, once placed, the final position error is only 4.2 mm and the final orientation error is only $3.6°$. The RMS position and orientation norm errors during the whole box placement motion amounts to 26.26 mm and $5.45°$ respectively. The same analysis is performed for the other boxes in the various sequences, for which the final pose errors are shown in Table 4.

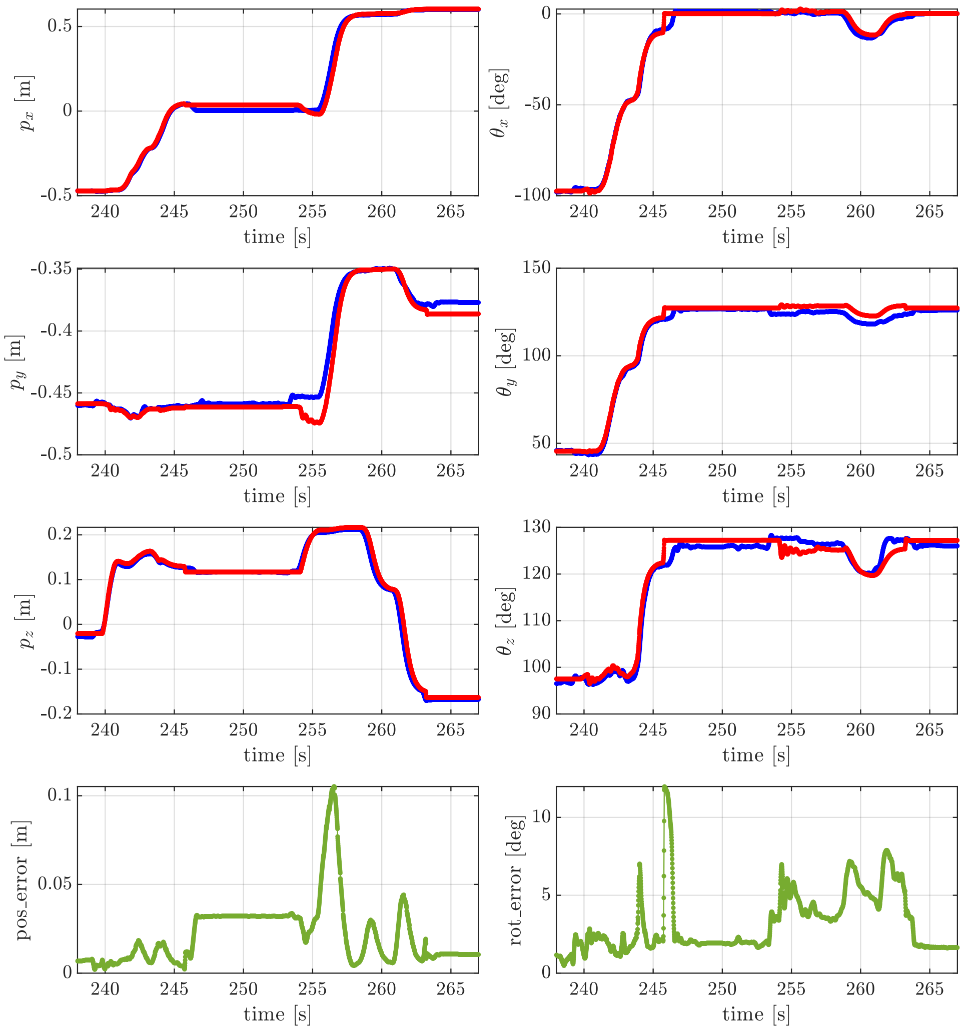


Figure 16: Resulting poses for the full motion of Box027 in Sequence 5.

The second motion of interests we are reporting here, is the one of Box027 in sequence 5, for which the evolution of the pose during manipulation is shown in Figure 16. This is a particularly complex motion which involves a flipping of the box itself, as also done for the motion described above, followed by a tight insertion between two other boxes within the container. At around $t = 239$ seconds, the box is picked-up from the conveyor and flipped on the workbench. The pose pattern for this phase of the motion is recognizably similar to the motion performed for Box016, this is especially true for the z-component of the position vector, despite, due to the plot scale, is not well visible in the plot of such component, but clearly visible on the position error norm plot. At $t = 254$ seconds, the box is re-picked from the workbench. From this time on, the box is brought on top of the container ($t = 258$ seconds), tilted such that the lower edge fits within the gap left open within two other boxes in the container ($t = 260$ seconds), and finally pushed down while re-aligning the box back straight vertical. In this way, the gap is further enlarged, freeing enough space for the box to be accommodated and finally released at $t = 265$ seconds. The final position error is 4.7 mm while the final orientation error is only $1.6°$. The RMS position and orientation norm errors during the whole box placement motion amounts to 28.21 mm and $4.02°$ respectively.

The pose of the boxes over time, during holding and interaction phases, is quite well captured by the simulation, meaning that the implemented physics models in AGX Dynamics are able to accurately describe the different phenomena in the experiment (contacts, friction, restitution, suction-cups elastic behavior and so on).

The executed motions will have consequences in the forces and torques sensed by the 6D force/torque sensor mounted at the robot's flange, as we will see in more detail in the evaluation of the wrench prediction.

Wrench predictions: In Figures 17 and 18, the wrenches of Box016 in sequence 1 and Box027 in sequence 5, respectively, are plotted over time. On the left column, we have shown the x, y, and z- components of the forces, and the norm of the force error over time. On the right, we have shown the torques around the x, y, and z axes over time. The time window corresponds to that of Figures 15 and 16 respectively, so a direct comparison can be made between the motions shown in the pose plots Figure 15 and 16 and the forces and torques related to them.

Before the execution of each sequence, a F/T sensor calibration procedure is performed to de-bias the sensor readings. Despite this, small offsets can still appear. The bias, slowly changes over time due to many factors, among which the temperature, and the time elapsed between two recordings of the same sequence can be beyond 15 minutes. Bias (offset) compensation is further tuned for each sequence for a fair comparison against simulation, which do not present (simulate) this phenomena.

Considering Figure 17, the first moment of interest is during the flipping motion between $t = 133$ and $t = 141$. Recall that the F/T sensor is placed between the gripper and the flange of the robot, meaning the forces and torques shown are also affected by the mass of the gripper. Note that this time window is longer than the time window for the flipping motion of the box, as after the box is placed, the robot is still moving the gripper back in vertical position. It is remarkable to see how the profile of the forces and torques during this motion is captured by the simulation. Especially knowing that during this phase, there are additional forces and torques generated by the mass of the object attached to the suction cup. This means that the implemented model of the suction cup is also able to closely capture the forces and torques applied to the system. At $t = 153$, a clear difference is visible between the forces and torques from simulation and the real experiment. This offset occurs due to the fact that the moment of impact between the box and the container, which happens during placement, is slightly different both in time

	Seq. 1		Seq. 2		Seq. 3		Seq. 4		Seq. 5	
	RMS_{pos} [mm]	RMS_{rot} [deg]	RMS_{pos} [mm]	RMS_{rot} [deg]	RMS_{pos} [mm]	RMS_{rot} [deg]	RMS_{pos} [mm]	RMS_{rot} [deg]	RMS_{pos} [mm]	RMS_{rot} [deg]
Box013	9.97	2.35	-	-	23.80	5.59	-	-	19.50	2.23
Box014	28.14	1.98	n.a.	n.a.	-	-	n.a.	n.a	n.a.	n.a.
Box015	11.57	3.79	16.67	3.46	-	-	24.52	6.00	9.84	4.54
Box016	26.26	5.49	-	-	38.93	4.02	-	-	16.25	1.39
Box017	12.19	3.45	7.00	2.43	6.67	2.85	22.09	3.09	7.83	3.25
Box018	15.53	2.18	9.99	2.42	7.58	1.74	7.30	1.81	8.46	2.25
Box019	-	-	8.42	2.63	23.78	6.33	41.87	3.24	50.37	3.33
Box020	-	-	-	-	55.96	4.09	21.92	4.20	-	-
Box021	-	-	-	-	-	-	48.97	3.61	-	-
Box022	-	-	-	-	-	-	n.a.	n.a.	-	-
Box023	29.53	3.00	26.40	2.55	46.03	5.17	19.53	2.12	31.15	2.92
Box024	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a	n.a.	n.a.	n.a
Box025	-	-	35.87	3.25	31.88	3.79	40.05	2.21	32.98	3.75
Box026	-	-	24.09	2.11	33.64	3.56	29.65	2.76	25.82	2.54
Box027	12.00	2.04	n.a.	n.a.	22.93	1.94	-	-	28.21	4.02
Box028	9.89	1.73	n.a.	n.a.	59.72	4.54	-	-	-	-
Box029	21.69	1.93	n.a.	n.a.	-	-	19.70	3.36	-	-
Box030	18.54	1.69	-	-	9.57	2.22	6.37	2.25	-	-

Table 5: RMS position and orientation prediction errors for the different boxes in each sequence. The dash indicates that the specific box is not part of the sequence. When a value is n.a. (not available) means that the Mocap system has lost track of the box, preventing us from recovering the values.

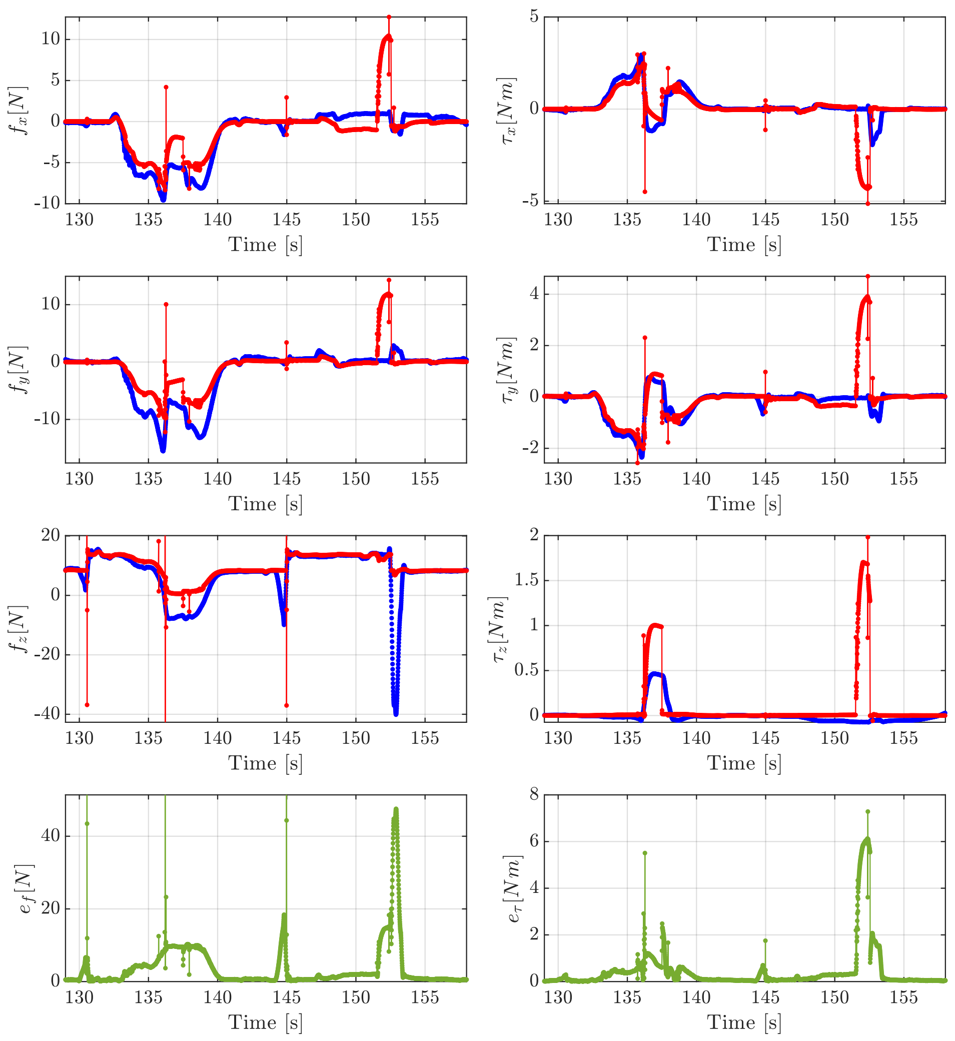


Figure 17: Resulting wrenches for the full motion of Box016 in Sequence 1.

and in space, and since no feedback is used, the controller cannot compensate for the small mismatch between the simulated and real environment. This type of information obtained from simulation can be used in the trajectory planning, knowing that during placement of the object, impacts are expected to occur. Also note that in the simulation, the full sequence has been run in open-loop, meaning that no data from experiments was used to update the poses of the boxes in simulation after placement. Given that Box016 is the seventh box in line of sequence 1, small deviations in the placement of previous boxes can lead to impacts in simulations that will not occur in experiments. Furthermore, the stiffness of the boxes in simulation is quite high compared to the real ones, causing large spikes in the simulation when making contacts (see also Figure 18 at $t = 239$ and $t = 254$ corresponding to the moments in which Box027 in sequence 5 is picked and re-picked up. In simulation, the size of the boxes correspond to the maximum size of the real box. The small difference in the box size (few mm) and the commanded pick pose causes a very high force on the z-axis). The force and torque RMS error norms are 7.75 [N] and 1.08 [Nm] respectively.

Considering now Figure 18, the first moment of interest is the flipping motion happening between $t = 238$ and $t = 250$. This phase is quite similar to the flipping of Box016 i sequence 1 and a similar analysis therefore applies. After re-picking Box027 from the workbench at $t = 254$, it is brought on top of the container and a tilting of the box to pre-insert the box within the gap is performed. Despite the tilting and the fact that the suction cup is in between, the shape and amplitude of the forces and torques signals are captured quite accurately. Some differences in the torques about the x and y axes between $t = 262$ and $t = 264$ is visible. This is due to the fact that in the real experiment, the manipulated box hits (twice) the already placed boxes in the container causing the arise of extra torque (see also the forces at the corresponding time window), which is not observed for the simulation since the latter does not make contact. Two spikes in the force along the z-axis are visible at the time instant the box is picked and re-picked up, as also happened for Box016. The force and torque RMS error norms are therefore 30.8 [N] and 0.99 [Nm] respectively.

The obtained results are quite exciting and astonishing, validating all the developed (and already existing) physics models in AGX dynamics (especially the suction cup model), paving the way for the use of the developed framework for advanced planning and manipulation control in the context of robotic process automation for logistics operations.

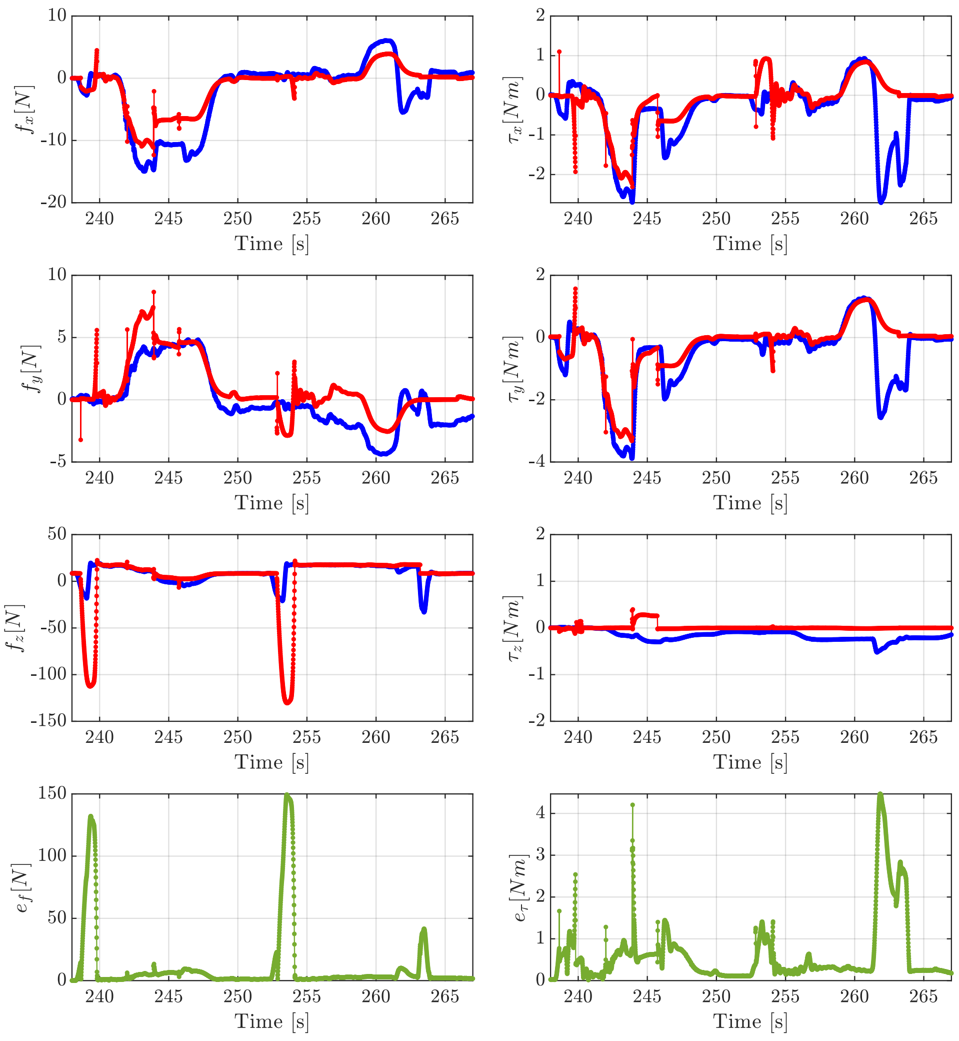


Figure 18: Resulting wrenches for the full motion of Box027 in Sequence 5.

	Seq. 1		Seq. 2		Seq. 3		Seq. 4		Seq. 5	
	RMS_f [N]	RMS_τ [Nm]	RMS_f [N]	RMS_τ [Nm]	RMS_f [N]	RMS_τ [Nm]	RMS_f [N]	RMS_τ [Nm]	RMS_f [N]	RMS_τ [Nm]
Box013	7.50	1.09	-	-	40.17	1.07	-	-	28.75	2.48
Box014	7.66	0.87	9.46	0.66	-	-	17.15	0.85	13.04	2.28
Box015	12.78	0.99	9.70	1.00	-	-	19.08	1.15	16.42	3.32
Box016	7.75	1.08	-	-	41.62	6.65	-	-	24.92	2.24
Box017	7.96	0.32	26.13	0.77	8.51	0.38	8.53	1.10	8.64	0.41
Box018	6.37	0.33	25.24	0.92	15.92	0.39	10.92	0.40	9.18	0.35
Box019	-	-	30.26	0.67	12.96	0.52	13.53	0.59	11.70	0.58
Box020	-	-	-	-	15.03	3.43	11.43	0.43	-	-
Box021	-	-	-	-	-	-	6.06	0.25	-	-
Box022	-	-	-	-	-	-	11.29	1.50	-	-
Box023	23.24	0.72	26.01	1.17	25.83	2.56	47.93	0.38	21.26	4.85
Box024	20.68	3.20	14.94	1.39	15.60	2.21	44.13	0.41	29.46	5.04
Box025	-	-	18.22	0.59	18.54	1.79	37.87	0.30	34.68	6.72
Box026	-	-	19.79	0.89	19.86	1.69	20.54	4.08	17.14	1.38
Box027	7.64	0.44	13.96	0.42	14.70	0.75	-	-	30.80	0.99
Box028	7.27	0.41	9.96	1.49	38.15	0.78	-	-	-	-
Box029	16.55	0.40	18.84	0.46	-	-	49.41	6.09	-	-
Box030	17.89	0.39	-	-	46.25	0.51	46.93	1.86	-	-

Table 6: RMS force and torque prediction errors for the different boxes in each sequence. The dash indicates that the specific box is not part of the sequence.

5 Performance testing

In this section, the performance of the BOX scenario with the addition of the developed components for I.AM., described in Section 2, is compared with the performance achievable by a human operator, as well as an example of a state-of-the-art pick-and-place robotic system: the Item Picker application, developed by Smart Robotics and deployed at their customers. This section defines the methods used for preparing and performing the comparative performance tests.

5.1 Test scenarios and procedures

Looking in more depth into the test plan, 3 test scenarios were devised:

- **Manual packing**
- **Robotic state-of-the-art packing**
- **Robotic packing with I.AM.**

These test scenarios will be explained in more detail in the coming subsections.

Across all test scenarios, the items are placed in the same container.

For each of these test scenarios, 5 item sequences (the order in which the items are supplied to the item placing agent) are provided. These sequences are the same for each of the test scenarios being considered, such that their performance relative to each other can be fairly compared. For each item sequence, the items provided are placed inside the container, one-by-one, until the item placing agent determines that no more items will fit inside. This ends the test, and the KPIs are calculated for that run. The same procedure is followed for the next item sequence, until all item sequences were tested.

5.1.1 Manual packing

For each item sequence, items are handed to a human operator who places the items in the container, one by one. The operator may flip items, rearrange items after placement, and allow items to touch. Items should not be visibly damaged by the placement (crushing, scratches). Items must be placed within the volume of the container (items may not stick out above the container). This scenario represents the best-possible achievable filling degree for a given sequence of items, and the container being considered. When comparing the filling degree of all test scenarios, this will be considered the highest possible achievable value, as reference.

5.1.2 Robotic state-of-the-art packing

The Smart Robotics Item Picker (SRIP) application is used as a representative of state-of-the-art item packing in the industry [12]. The same sequences are presented to the robot in the same order as in 5.1.1. This is an automated system, and it will autonomously pick the item, infer its dimensions and place it in the container at a pose chosen by its packing algorithm. It has no knowledge of the next items to pick, only of the item being processed and the items already placed in the container.

The SRIP test setup is located in the Smart Robotics Innovation Center in Best, The Netherlands. The test setup is similar to the one described in Section 3.2. It consists of the following relevant components:

- A Universal Robots UR10 cobot, used as the manipulation agent that will interact with the boxes during packing. It is equipped with a Smart Robotics GS002 vacuum gripper, with a Piab piGRIP 50mm diameter suction cup, with a flexible lip. The robot will be controlled using the Smart Robotics Item Picking application (proprietary software).
- A 570x370x250 mm container, where the items will be picked from (the source container).
- A 570x370x250 mm container, where the items will be placed (the target container).
- An Ensenso 3D vision camera, looking directly into the target container, to capture the state of the container before and after each place.
- A Realsense 2 depth camera, used to estimate item dimensions before each place.
- A video camera placed above the target container to visualize how the target container is filled.

The SRIP application is configured to wait for a request to pick an item from the source container and place it in the target container. A user manually sends this request to the application after placing the next box to be picked in the item sequence (described in Section 3.1.2) into the source container. This allows the item sequence to be strictly adhered to.

The SRIP is not capable of flipping the item before placement in this configuration.

The application has been configured to leave a 2cm margin between items placed and between the outer edge of the target container. This is to account for items potentially swinging during placement, which can result in a collision.

The test procedure for each item sequence is the following (see Figure 19 for flowchart):

1. The next item in the sequence is placed into the source container.
2. A pick is requested.
3. The robot attempts to place the item into the target container.
4. Repeat from step 1 until no more items can be placed by the application or there are no more items listed in the sequence.
5. Data from the robot's operation collected by the application, as well as depth and RGB images of the target container, are stored for later analysis.

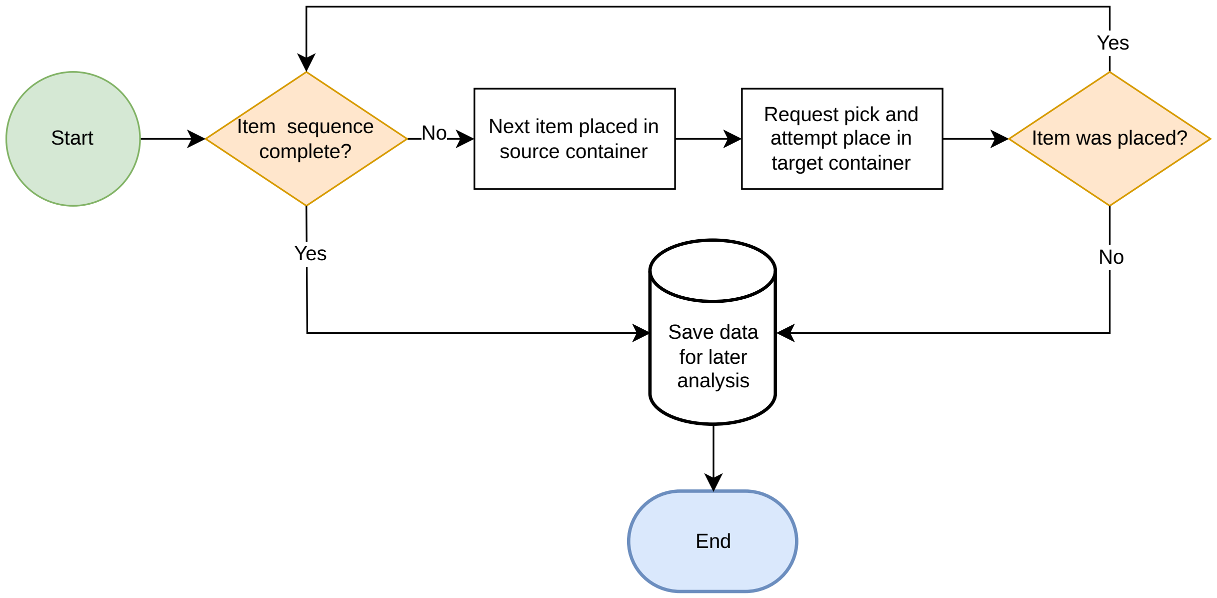
The filling degree KPI specified in Section 1.4 is computed based on the volume of the items that were successfully placed in the container.

5.1.3 Robotic packing with I.AM.

The setup described in Section 3.2 is used for performing the I.AM. BOX test.

The procedure used when testing is the same as previously presented in Section 4.3.

The Filling Degree KPI specified in Section 1.4 is computed based on the volume of the items that were successfully placed in the container.



Start
Item sequence complete?
No
Next item placed in source container
Request pick and attempt place in target container
Item was placed?
Yes
Yes
No
Save data for later analysis
End

Figure 19: Smart Robotics Item Picker test procedure flowchart.

5.2 Filling degree results and evaluation

Table 7 aggregates the filling degrees achieved by each test scenario, for each test sequence.

	Seq. 1	Seq. 2	Seq. 3	Seq. 4	Seq. 5
State of the Art	32.21 %	41.41 %	41.18 %	29.55 %	39.59 %
I.AM.	75.66 %	67.45 %	70.00 %	68.00 %	67.98 %
Manual Operator	75.66 %	67.45 %	70.00 %	73.68 %	77.92 %

Table 7: Filling degree achieved by each test scenario, for each test sequence.

The manual operator scenario (explained in Section 5.1.1) represents the maximum filling degree achievable for the given items, container and item sequences. As expected, the state-of-the-art system is very conservative in avoiding collisions and consequently achieves a much lower filling degree than what the manual operator can achieve. This difference is not always the same, and heavily depends on the size of the items and the container, as well as the specific constraints of the system. In this case, the system used is not able to flip items in order to place them in a more advantageous orientation - being limited to just rotating the items around the vertical axis. The I.AM. demonstrator, on the other hand, achieves filling degree values comparable to the manual operator scenario - partially due to its ability to rotate items in more than one axis using a flip motion (see Figure 13). Although the place poses and motion waypoints were programmed by a human, it shows that with good enough modeling, vision system, motion planner and packing algorithm, performance comparable to a human operator should, in practice, be achievable.

6 Conclusion

Industrial robotic applications can benefit from a contact-rich interaction with their environment. More specifically, item packing robotic applications could potentially pack more items in the same container volume. Consequently, transportation and operational costs are reduced when compared to the current robotic state of the art solutions, in turn making the adoption of robotic solutions employing this technology for such applications more viable and financially appealing.

The objective of the BOX demonstrator is to show that a simulation environment may be constructed which accurately forecasts not only the position and orientation of boxes during complex manipulation tasks with a vacuum gripper (e.g., flipping and placement into a container) but also the interaction forces involved, as deemed relevant by Smart Robotics and Vanderlande for the e-commerce industry. This simulation environment has been implemented with the RACK framework and validated to reflect reality with impressive levels of accuracy as it was shown in Section 4.4. More specifically, the obtained results for various challenging motions of flipping and stacking boxes during multi-contact situations show a good match between simulation and real experiments, thereby validating the developed models within the I.AM. project and implemented in AGX dynamics (e.g., the suction cup model).

On the other hand, Section 5 shows that combining the I.AM.-developed simulation environment with (at the moment nonexistent) sufficiently intelligent motion and packing algorithms could yield packing density performance (filling degree) far superior to the current state of the art and comparable to that of a human operator as was summarized in Table 7. In Figure 20, a comprehensive overview of the achievable filling degree by a human operator, current state of the art robotic system and I.AM.-developed technology is shown for all the 5 executed sequences. This encouraging result provides motivation for further research into contact-rich and contact-aware motion planning, and packing algorithms that also exploits in an intelligent manner the environment. In the long-run, this may facilitate the adoption of industrial robotic applications in the e-commerce industry.

The developed technology paves the way for the use of this framework for advanced planning and manipulation control in the context of robotic process automation for logistics operations.

7 References

[1] B. Coenen and S. de Looijer, "Deliverable D7.3: Roadmap and business cases for introduction of I.AM technology on logistics," H2020 EU Project I.AM. Consortium, Tech. Rep. D7.3, 2024.

[2] M. J. Jongeneel, A. Oliva, J. den Ouden, A. Saccon, and J. van Steen, "Deliverable D1.3: I.Model Report," H2020 EU Project I.AM. Consortium, Tech. Rep. D1.3, 2023.

[3] A. 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.

[4] A. Kurdas, M. Hamad, J. Vorndamme, N. Mansfeld, S. Abdolshah, and S. Haddadin, "Online payload identification for tactile robots using the momentum observer," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 5953–5959.

[5] P. Gergondet, C. Lacoursière, F. Nordfeldth, and J. van Steen, "Deliverable D1.2: Physics Engine API," H2020 EU Project I.AM. Consortium, Tech. Rep. D1.2, 2023.

[6] T. Bosch *et al.*, "Deliverable D5.3: Scenario 1 (TOSS) report," H2020 EU Project I.AM. Consortium, Tech. Rep. D5.3, 2022.

[7] CNRS-AIST JRL, CNRS LIRMM, "mc_rtc," 2020. [Online]. Available: https://jrl-umi3218.github.io/mc_rtc/index.html

[8] P. Gergondet, A. Kheddar, A. Saccon, J. van Steen, and A. Zermane, "Deliverable D4.1: I.Control Report," H2020 EU Project I.AM. Consortium, Tech. Rep. D4.1, 2024.

[9] C. F. Henst, "Quantifying the Sim-to-Real Gap in Robotic Bin Packing via Nonsmooth Physics Simulation," Master's thesis, Eindhoven University of Technology, Faculty of Mechanical Engineering, Dynamics and Control Group, February 2024, dC 2024.010.

[10] M. J. Jongeneel, J. den Ouden, and A. Saccon, "Deliverable D1.4: Publication of I.AM. Dataset," H2020 EU Project I.AM. Consortium, Tech. Rep. D1.4, 2022.

[11] M. J. Jongeneel, S. Dingemans, and A. Saccon, "The Impact-Aware Robotics Database: Supporting Research Targeting the Exploitation of Dynamic Contact Transitions," *Submitted to: IEEE Robotics and Automation Letters (RA-L)*, August 2023.

[12] Smart Robotics, "Smart Item Picker," 2023. [Online]. Available: https://smart-robotics.io/item-picking/

A Item sets

This appendix contains the detailed information about the properties of the boxes and the five item sequences.

Box Type	Size x $[mm]$	Size y $[mm]$	Size z $[mm]$	Mass $[kg]$
B1	200	200	100	0.55
B2	105	120	110	0.26
B3	185	260	60	1.05
B4	170	235	95	0.95

Table 8: Nominal size and mass of the representative items from which an item set was realized (Section 3.1.1).

Item ID	Box Type	Size x $[mm]$	Size y $[mm]$	Size z $[mm]$	Mass $[kg]$
BOX013	B1	203	198	97	0.550
BOX014	B1	200	200	97	0.547
BOX015	B1	201	202	97	0.550
BOX016	B1	200	201	96	0.547
BOX017	B2	106	121	107	0.257
BOX018	B2	104	119	108	0.265
BOX019	B2	104	119	109	0.260
BOX020	B2	105	121	108	0.262
BOX021	B2	105	118	107	0.260
BOX022	B2	105	120	108	0.262
BOX023	B3	184	259	62	1.045
BOX024	B3	185	259	61	1.026
BOX025	B3	184	259	62	1.059
BOX026	B3	183	259	62	1.049
BOX027	B4	170	234	94	0.946
BOX028	B4	170	236	94	0.948
BOX029	B4	170	235	93	0.954
BOX030	B4	171	233	94	0.948

Table 9: Size and mass of the items composing the item set.

Item ID	Box Type	$I_{xx}\ [kg \cdot m^2]$	$I_{yy}\ [kg \cdot m^2]$	$I_{zz}\ [kg \cdot m^2]$
BOX013	B1	0.00211477	0.0021974	0.0038231
BOX014	B1	0.00213103	0.00213103	0.00378219
BOX015	B1	0.00217197	0.00215617	0.00384852
BOX016	B1	0.00214394	0.00212715	0.00380181
BOX017	B2	0.000369	0.00035505	0.00057869
BOX018	B2	0.00040096	0.00037402	0.00060402
BOX019	B2	0.00038184	0.00036295	0.00058612
BOX020	B2	0.00038885	0.00036729	0.00059513
BOX021	B2	0.00038052	0.00036187	0.00058742
BOX022	B2	0.00038861	0.00036745	0.0005948
BOX023	B3	0.0059138	0.00325435	0.00864822
BOX024	B3	0.00574403	0.00318144	0.00843208
BOX025	B3	0.00604828	0.00331967	0.00883353
BOX026	B3	0.00596339	0.00326071	0.00869884
BOX027	B4	0.00456245	0.00239275	0.00652735
BOX028	B4	0.00460268	0.00239628	0.00656807
BOX029	B4	0.00462445	0.00242993	0.0066061
BOX030	B4	0.00455881	0.00241266	0.00653819

Table 10: Principal moments of inertia I, computed around the center of mass, for all items of the test set.

Sequence 1	Sequence 2	Sequence 3	Sequence 4	Sequence 5
B4 (BOX027), Z↑	**B3** (BOX024), X↑	**B3** (BOX024), X↑	**B2** (BOX017), Z↑	**B1** (BOX013), Z↑
B4 (BOX028), Z↑	**B3** (BOX025), X↑	**B2** (BOX017), Z↑	**B3** (BOX023), Z↑	**B1** (BOX016), Z↑
B1 (BOX013), Z↑	**B3** (BOX026), X↑	**B2** (BOX018), Z↑	**B2** (BOX018), Z↑	**B3** (BOX026), X↑
B1 (BOX014), X↑	**B4** (BOX027), X↑	**B4** (BOX030), Z↑	**B4** (BOX030), Z↑	**B1** (BOX014), X↑
B2 (BOX017), Z↑	**B4** (BOX028), X↑	**B1** (BOX013), Z↑	**B3** (BOX024), Z↑	**B3** (BOX023), X↑
B1 (BOX015), X↑	**B4** (BOX029), X↑	**B4** (BOX027), Z↑	**B3** (BOX025), Z↑	**B3** (BOX024), X↑
B1 (BOX016), Z↑	**B2** (BOX017), Z↑	**B2** (BOX019), Z↑	**B2** (BOX019), Z↑	**B3** (BOX025), X↑
B4 (BOX029), Z↑	**B2** (BOX018), Z↑	**B3** (BOX025), X↑	**B1** (BOX014), X↑	**B2** (BOX017), Z↑
B2 (BOX018), Z↑	**B1** (BOX014), X↑	**B3** (BOX026), X↑	**B3** (BOX026), X↑	**B1** (BOX015), X↑
B4 (BOX030), Z↑	**B1** (BOX015), X↑	**B1** (BOX016), Z↑	**B2** (BOX020), Z↑	**B2** (BOX018), Z↑
B3 (BOX023), Z↑	**B2** (BOX019), Z↑	**B3** (BOX023), X↑	**B4** (BOX029), Z↑	**B4** (BOX027), X↑
B3 (BOX024), Z↑	**B3** (BOX023), X↑	**B2** (BOX020), Z↑	**B1** (BOX015), X↑	**B2** (BOX019), Z↑
B2 (BOX019), Z↑	**B1** (BOX016), Z↑	**B4** (BOX028), Y↑	**B2** (BOX021), Z↑	**B4** (BOX028), Z↑
B2 (BOX020), Z↑	**B1** (BOX017), Z↑	**B1** (BOX014), Z↑	**B2** (BOX022), Z↑	**B2** (BOX020), Z↑
B3 (BOX025), Z↑	**B4** (BOX030), Z↑	**B2** (BOX021), Z↑	**B4** (BOX027), Z↑	**B2** (BOX021), Z↑
B3 (BOX026), Z↑	**B2** (BOX020), Z↑	**B1** (BOX015), Z↑	**B1** (BOX013), Z↑	**B4** (BOX029), X↑
B2 (BOX021), Z↑	**B2** (BOX021), X↑	**B2** (BOX022), Z↑	**B1** (BOX016), Z↑	**B4** (BOX030), X↑
B2 (BOX022), Z↑	**B2** (BOX022), Z↑	**B4** (BOX029), Z↑	**B4** (BOX028), Z↑	**B2** (BOX022), Z↑

Table 11: Item sequences used for testing. **Bx** represents the box type of the items that should be fed to the robot, in that particular order. The unique box ID assigned to each individual box are in parenthesis. The letter next to the arrow pointing up (X, Y, or Z) indicates the axis of the box that should point up (see Figure 7), at the pick location. Yaw is considered irrelevant, since the robot can easily rotate around the Z axis of the box to place it at the desired yaw rotation.

B Packing Results

This appendix contains an overview of the packing results for manual packing, packing with current state of the art, packing with I.AM. technology (experiments), and packing with I.AM. technology (simulation).

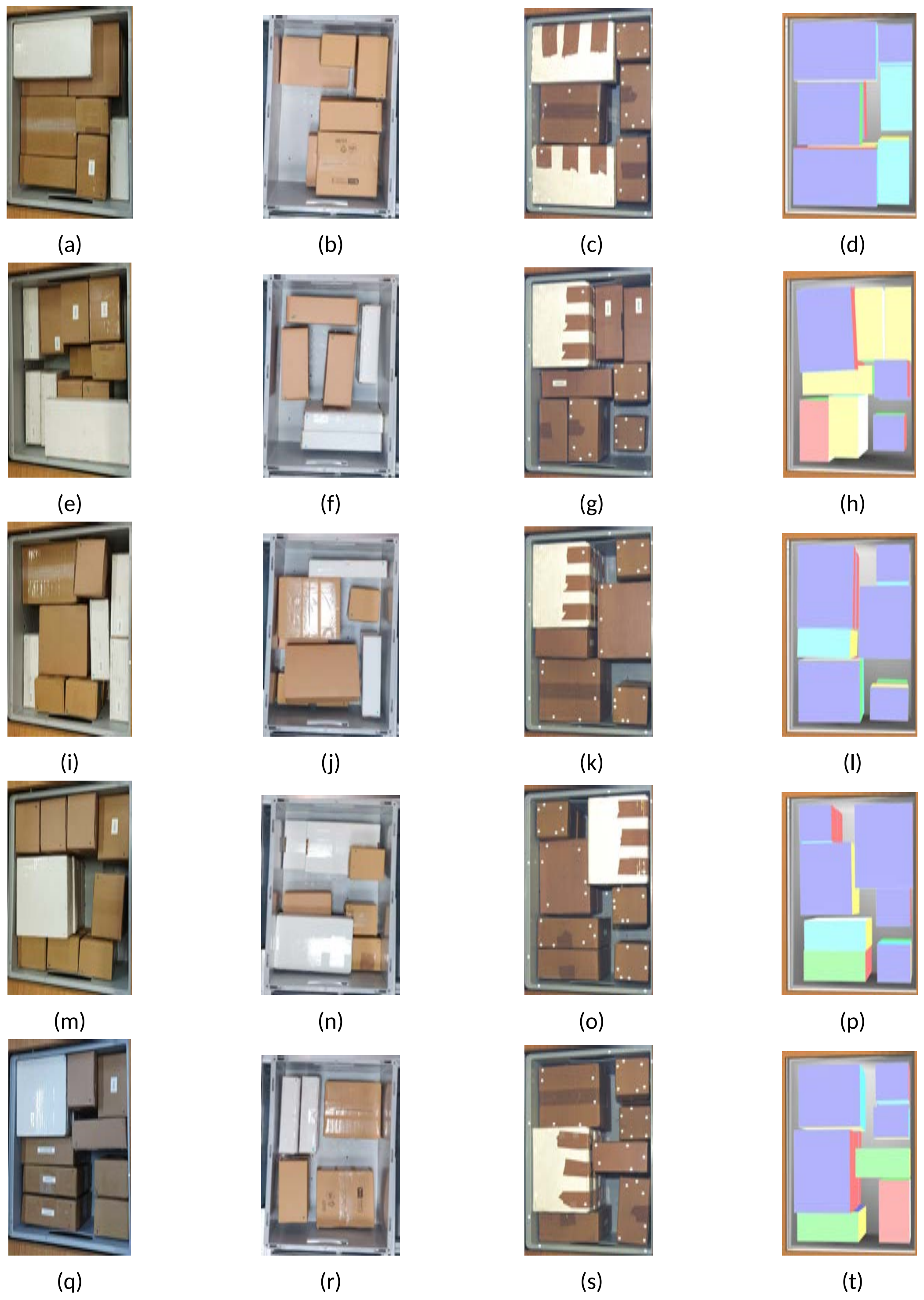
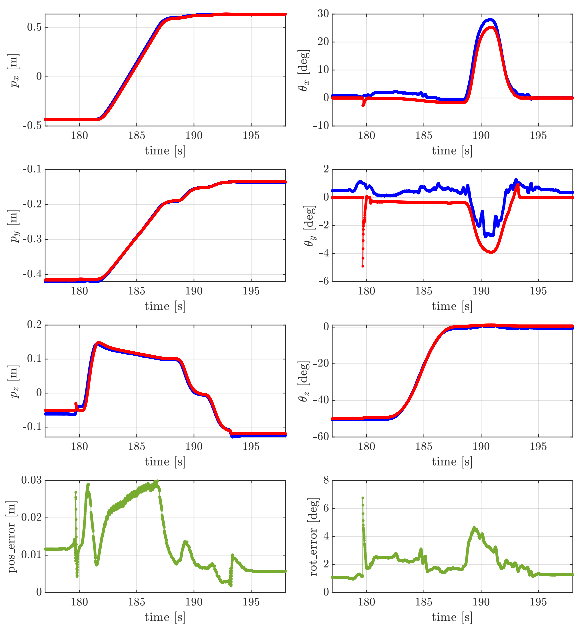
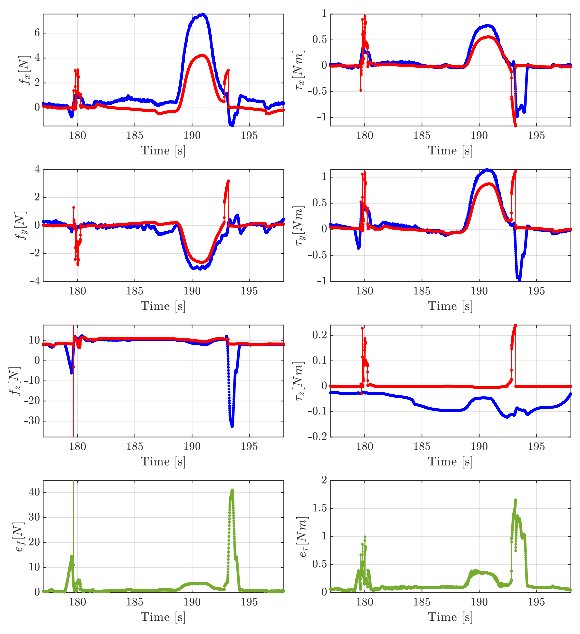


Figure 20: Sequences packed using different methods. Each row represents a sequence (1 through 5) and each of the first three columns represents a packing method. From left to right: human operator, autonomously with a state-of-the-art robotic system (Smart Robotics Item Picker) and autonomously by the I.AM. experimental setup. The last column shows the final state of the container in the simulation using AGX Dynamics (compare with I.AM. experimental setup state).

C Prediction results during key phases

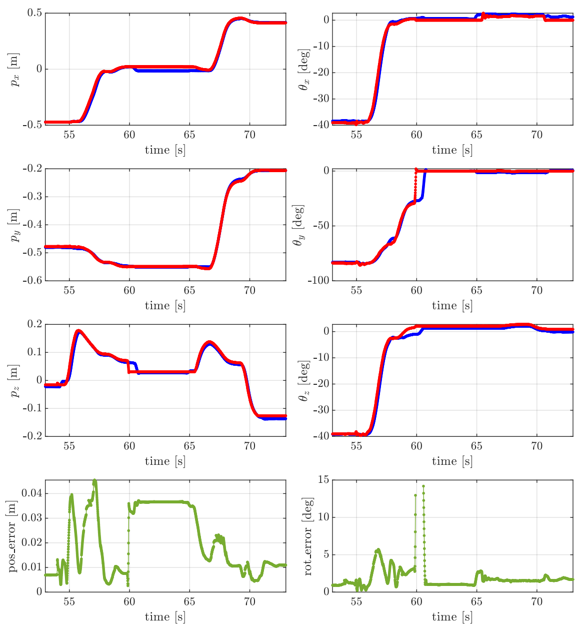


(a)



(b)

Figure 21: Box018 in Sequence 1. (a) x, y and z components of the position and rotation vector of the box for the simulation (——) and reconstructed mocap data (——). (b) x, y and z components of the sensed forces and torques in simulation (——) and real data (——).


(a)

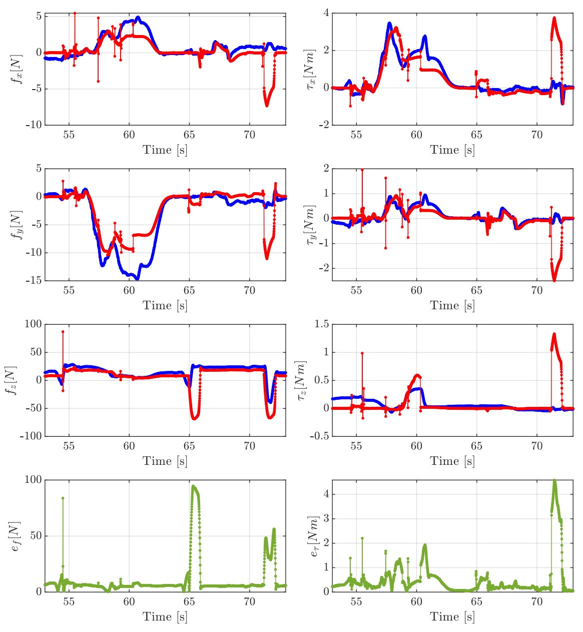
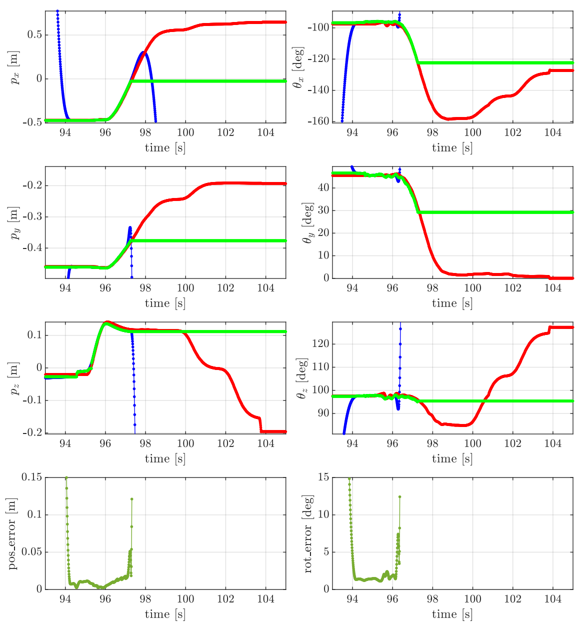
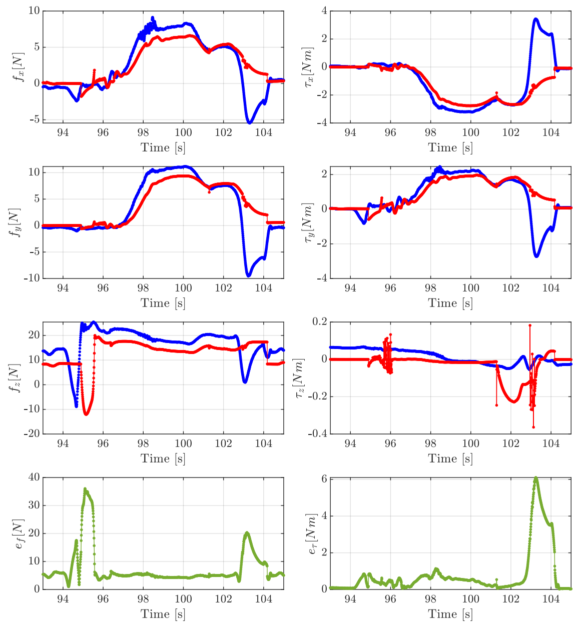

(b)

Figure 22: Box026 in Sequence 2. (a) x, y and z components of the position and rotation vector of the box for the simulation (——) and reconstructed mocap data (——). (b) x, y and z components of the sensed forces and torques in simulation (——) and real data (——).

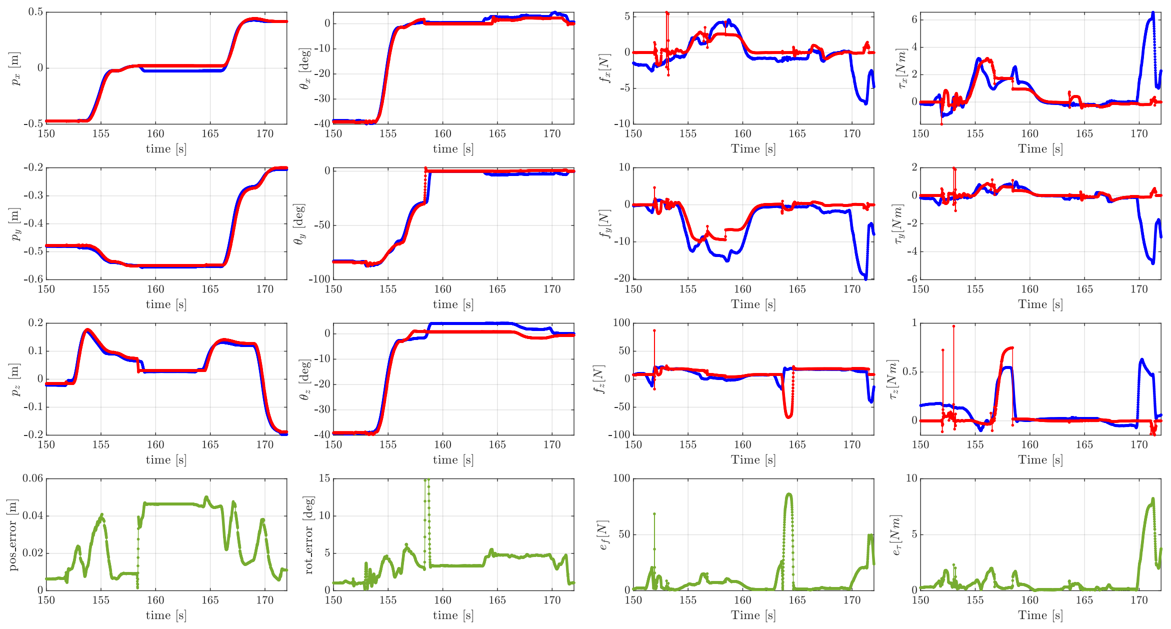


(a)



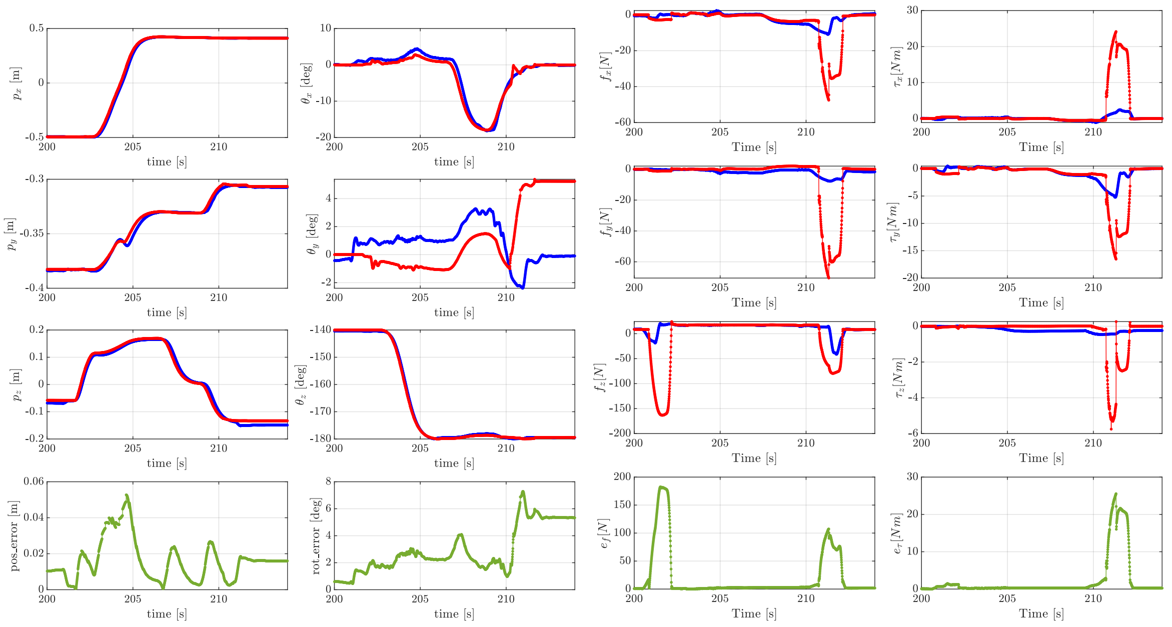
(b)

Figure 23: Box028 in Sequence 2. (a) x, y and z components of the position and rotation vector of the box for the simulation (——), mocap raw data (——) and reconstructed data (——). (b) x, y and z components of the sensed forces and torques in simulation (——) and real data (——). The Mocap system loosed tracking of the box after $t > 97.2$ [s] affecting the positional and rotational RMS errors as well as the final pose values which are lost.



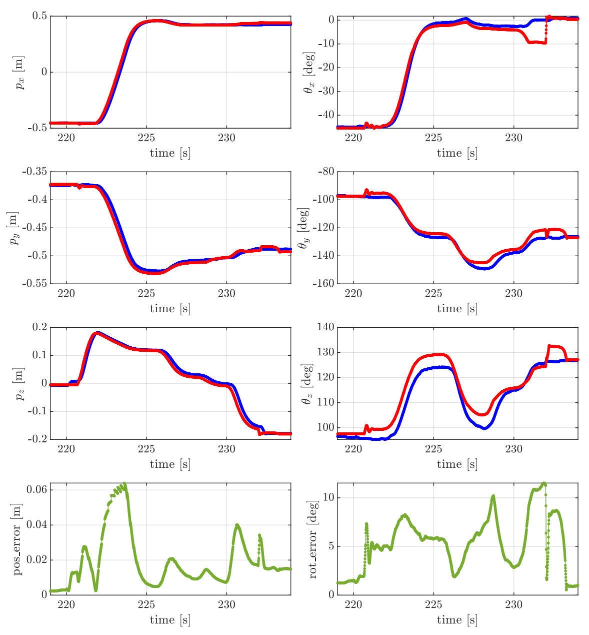
(a) (b)

Figure 24: Box025 in Sequence 3. (a) x, y and z components of the position and rotation vector of the box for the simulation (——) and reconstructed mocap data (——). (b) x, y and z components of the sensed forces and torques in simulation (——) and real data (——).

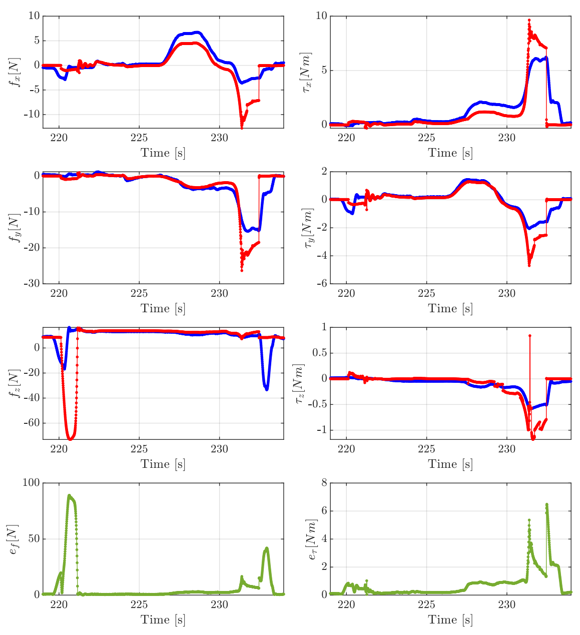


(a) (b)

Figure 25: Box029 in Sequence 4. (a) x, y and z components of the position and rotation vector of the box for the simulation (——) and reconstructed mocap data (——). (b) x, y and z components of the sensed forces and torques in simulation (——) and real data (——).

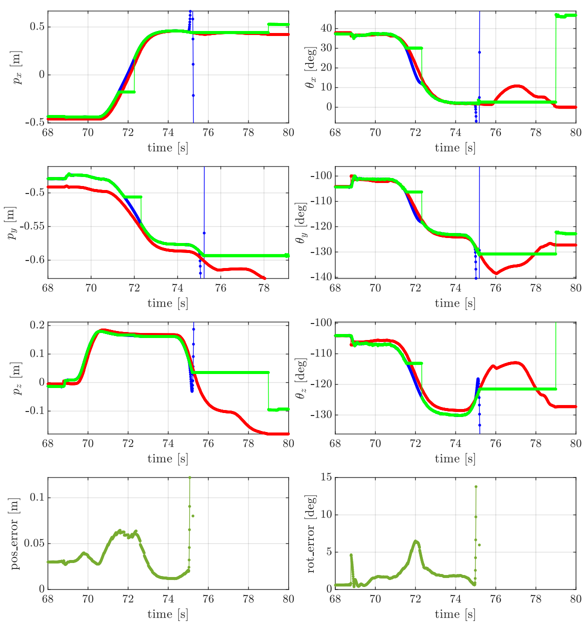


(a)

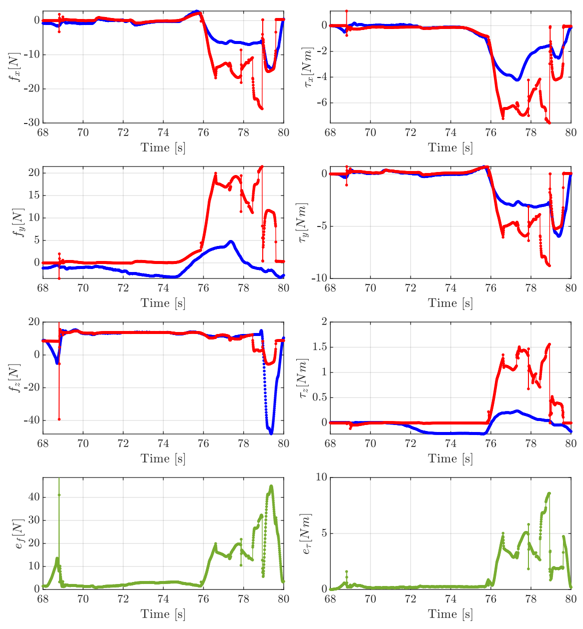


(b)

Figure 26: Box015 in Sequence 4.(a) x, y and z components of the position and rotation vector of the box for the simulation (—) and reconstructed mocap data (—). (b) x, y and z components of the sensed forces and torques in simulation (—) and real data (—).



(a)



(b)

Figure 27: Box014 in Sequence 5. (a) x, y and z components of the position and rotation vector of the box for the simulation (——), mocap raw data (——) and reconstructed data (——). (b) x, y and z components of the sensed forces and torques in simulation (——) and real data (——). The Mocap system loosed tracking of the box in $71.5 < t < 72.5$ [s] and for $t > 75.5$ [s] affecting the positional and rotational RMS errors as well as the final pose values which are lost.