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

Publication of I.AM. Dataset

Dissemination level	Public (PU)
Work package	WP1: Modeling
Deliverable number	D1.1
Version	F – 1.0
Submission date	30/09/2020
Due date	30/09/2020

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**
Maarten JONGENEEL	TU/e	m.j.jongeneel@tue.nl
Jos DEN OUDEN	TU/e	j.h.v.d.ouden@tue.nl
Alessandro SACCON	TU/e	a.saccon@tue.nl

Control sheet

Version history			
Vers.	**Date**	**Modified by**	**Summary of changes**
0.1	13/05/2020	Jos DEN OUDEN	ToC & first contents
0.11	01/07/2020	Maarten JONGENEEL	Document structure created
0.12	26/07/2020	Maarten JONGENEEL	Start of Chapters 2 and 3
0.13	03/09/2020	Alessandro SACCON	Given a Why/What/How/When/Who structure
0.14	03/09/2020	Abderrahmane KHEDDAR	Preliminary feedback
0.15	07/09/2020	Alessandro SACCON	Data structure further detailed
0.16	08/09/2020	Maarten JONGENEEL	Chapter 4 started
0.17	21/09/2020	Alessandro SACCON	Chapter 2 completed
0.18	27/09/2020	Maarten JONGENEEL	Chapter 4 completed
0.19	28/09/2020	Alessandro SACCON	Chapter 3 completed
0.5	28/09/2020	Alessandro SACCON	Pre-final version ready for peer-review
0.9	30/09/2020	Alessandro SACCON	Peer-review comments addressed
1.0	30/09/2020	Alessandro SACCON	Revised version ready for submission, quality check

Peer reviewers		
	Reviewer name	**Date**
Reviewer 1	Abderrahmane KHEDDAR	29/09/2020
Reviewer 2	Jos DEN OUDEN	29/09/2020

Legal disclaimer

TABLE OF CONTENTS

ABBREVIATIONS

Abbreviation	Definition
CAD	Computer Aided Design
CoM	Center of Mass
CSV	Comma Separated Value
EC	European Commission
HDF5	Hierarchical Data Format (version) 5
Mocap	Motion capture
PU	Public
RGB	Red Green Blue
RGBD	Red Green Blue Depth
STL	Standard Triangle Language
UR	Universal Robots
WP	Work Package

EXECUTIVE SUMMARY

This deliverable D1.1 Publication of I.AM. Dataset, aims at providing an overview of the current effort to create a data repository generated in WP1 (I.Model), that is to be used by the entire consortium. This deliverable builds upon the D6.3 Data Management Plan and consequently on the H2020 Online Manual on Data Management Plan and deals with the topics of FAIR data. The deliverable provides an in-depth description of the repository aim and structure, providing a concrete example represented by the storage of the first box-conveyor impacts recordings (approximately 500 recordings). Further detailed technical implementation of the repository will be described in D1.4 "Publication of I.AM. dataset (update D1.1)" at M36 (December 2023), where we will explain the whole content of the repository, its final structure, and the open access.

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 hardware-bearable impacts, while providing accurate sensing and actuation capabilities. I.AM. leverages this technology and strengthens European leadership by endowing robots to exploit intentional impacts for manipulation. I.AM. focuses on impact-aware manipulation in logistics, a new area of application for robotics which will grow exponentially in the coming years, due to socio-economical drivers such as booming of e-commerce and scarcity of labour.
I.AM. relies on four scientific and technological research lines that will lead to breakthroughs in modeling, sensing, learning and control of fast impacts:

- 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;
- I.Learn provides advances in planning and learning for generating desired control parameters based on models of uncertainties inherent to impacts;
- 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;
- I.Control generates a framework that, in conjunction with the realistic models, advanced planning, and sensing components, allows for robust execution of dynamic manipulation tasks.

This integrated paradigm, I.AM., brings robots to an unprecedented level of manipulation abilities. By incorporating this new technology in existing robots, I.AM. enables shorter cycle time (10%) for applications requiring dynamic manipulation in logistics. I.AM. will speed up the take-up and deployment in this domain by validating its progress in three realistic scenarios: a bin-to-belt application demonstrating object tossing (TOSS), a bin-to-bin application object fast boxing (BOX), and a case depalletizing scenario demonstrating object grabbing (GRAB).

1.2. Purpose of the deliverable

This deliverable D1.1 Publication of I.AM. dataset is a document summarizing (i) information about the collected datasets, i.e., the type of recorded motions, (ii) storage of the repository, i.e. the repository, infrastructure and data formats, and (iii) the taxonomy of the various impacts.
Deliverable D1.1 paves the way to reaching the following milestones:

- MS3 “Object-environment impact laws” – M18 (June 2021)
- MS8 “Scenario 1 (TOSS)” – M24 (December 2021)

- MS9 "Robot-environment impact laws" (TU/e) – M30 (June 2022)
- MS13 "Scenario 2 (BOX)" (Smart Robotics) – M36 (December 2022)
- MS14 "Robot-object-environment impact laws" (TU/e) – M42 (June 2023)
- MS15 "Impact motion learning" (EPFL) – M42 (June 2023)
- MS18 "Scenario 3 (GRAB)" (EPFL) – M42 (June 2023)

A final update of the I.AM. data repository will be provided in the deliverable D1.4 "Publication of I.AM. dataset (update D1.1)" at M36 (December 2023). The final version of the dataset will contain object-environment, robot-object, and robot-object-environment impact/release data, strictly related to the three validation scenarios (TOSS, BOX, GRAB). We also plan to pair the final version of the I.AM. data repository with a scientific paper to which it links.

Earlier, related to this deliverable, the following deliverables were published: deliverables D6.3 "Data Management Plan" (which will be updated in M25 as D6.5 and in M40 as D6.6) and related deliverables on ethics, being D8.1 "H - Requirement no.1" and D8.4 "POPD – Requirement no.2". These topics of FAIR data and data privacy is already described in these deliverables and therefore the reader is asked to read these documents for further details on these topics.

1.3. Intended audience

The dissemination level of D1.1 is 'public' (PU) – meant for members of the Consortium (including Commission Services) and the general public.

2. I.AM. DATA REPOSITORY

This section provides the motivation for the I.AM. data repository, its intended content and target audience. Furthermore, a preliminary vision for the repository's data structure, data collection, storage, and access are given.

2.1. Purpose of the I.AM. data repository (why)

Impact-aware manipulation is a growing field within the robotics field and the I.AM. project aims at providing a data repository with the following two purposes:

1. Store selected impact and release motion archives that are useful to develop the four key components that forms the impact-aware manipulation (I.AM.) technology, namely
   - Modeling (I.Model)
   - Learning and Planning (I.Learn)
   - Sensing (I.Sense)
   - Control (I.Control).

2. Make these impact-and-release-motion archives, that collectively form the I.AM. repository, available to other research institutions and companies worldwide, with the long-term goal of stimulating research and thus advancement in the field of impact-aware robotic manipulation.

The repository will target, with special attention, the storage of impact and release motions that are essential in the execution of the TOSS, BOX, GRAB scenarios of the project. Additionally, the repository will allow for storage of motions with no contact transitions (free motion/constrained motions) to allow for dynamic identification of specific subcomponents such as, e.g., the bellowed suction gripper used in the TOSS and BOX scenarios.

The long-term vision is to allow for the upload of external new archives, each containing selected recordings of impact-and-release motions, related to impact-aware manipulation experiments. The archives will be requested to be complying to a to-be-defined repository policy to guarantee re-use and searchability of the stored data, ensuring in particular that the stored data will have a value for the whole robotics community to developed, test, and compare methods related to modeling, learning, sensing, and control of impact-aware robotics systems.

2.2. Data to be collected (what)

The I.AM. data repository will contain recordings of impact-and-release motions involving known objects, robots, and well described environments. It will contain, in particular, time series data related to

- object impacting the environment
- objects impacting robots
- objects impacting the environment while held by robots
- objects impacted by robots while being in contact with the environment

and corresponding release motions

- objects detaching from the environment
- objects separating from a robot (in particular, its end effector)
- objects separating from the environment while held by robots
- objects separating from a robot while being in contact with the environment

A visual representation of the release and impact motions that are core to the I.AM. project, being related to the three validation scenarios, is given in Figure 1.

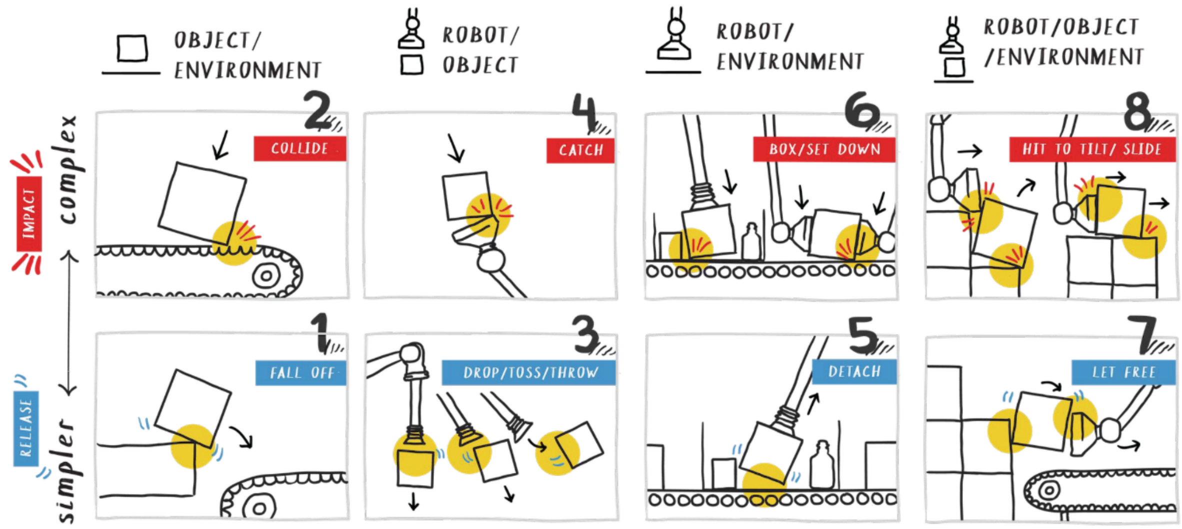

Figure 1: The qualitative I.AM. complexity metric used to cluster dynamic contact transitions in terms of increasing complexity and in terms of autonomous execution.

Each dataset of the I.AM. repository, focusing on one specific type of release/impact motion and scenario, will contain multiple recordings of time-synchronized times series data, collected by means of (commercially available) sensors and robots. A representative list of sensors, by which time series data that will be collected, is the following:

- robot joint displacement sensors (digital encoders/resolvers),
- robot joint torque sensors,
- robot joint current,
- force/torque sensors,
- standard cameras / depth cameras,
- motion capture systems,
- high-speed cameras / event cameras,

- accelerometers / IMU sensors,
- pressure sensors
- robot input commands,
- any other sensors attached to objects or the environment, when required.

It is important to classify the time series data based on their primary purpose. In particular, it is essential to distinguish between two types of time series data: (1) time series data relevant for online learning, sensing, estimation, and feedback control; (2) time series data relevant for offline modelling and learning. Concrete examples of these two categories are the following: (1) joint encoder and joint torque sensor data recorded during intentional-impact experiments with expected and unexpected contact-state outcome, to be used for causal online detection and classification (I.Sense); (2) motion capture times series data used offline for non-causal data processing, for modelling and learning predictive impact models/laws (I.Model). A third type of times series data is that used for enhancing human understanding of the recordings: the concrete example is a camera recording of impact/release experiments that is synchronized with the time series data obtained by other sensors.

Together with time series data and generic data (such as exact robot and sensor models, experimental conditions), metadata will also be included in each dataset, to allow for both human and machine interpretability and search. This includes many details such as the date and time when the experiment was conducted, and identifiers for the robot that was used along with the controller type and revision identifiers, and information about the experiment.

An archive will contain several recordings of impact/release motions of the same type, with the same or on-purpose different initial conditions, inputs, and feedback gains, with the aim of assessing repeatability/variability of the impact/release motion outcome of the experiments based on nominally identical conditions as well as exploring the functional relationship between the outcome as a function of the initial conditions.

2.3. Data collection, structure, and access (how)

One specific goal of the I.AM. project is to create an open-access data repository that will contain object-environment, robot-object, and robot-object-environment impact/release motion data. Having the I.AM. consortium agreed to join the H2020 Open Research Data Pilot, a preliminary Data Management Plan (DMP) has been released as D6.3 in June 2020, that has extensively covered the aspects of making the data FAIR (findable, openly accessible, interoperable, re-usable) and we therefore refer to D6.3 DMP for FAIR data related aspects. The focus in this section is on providing additional information with respect to D6.3 regarding how data will be collected, organized for storage, and possible options for it to be accessed remotely.

One specific aspect to consider is that there is currently no public repositories to store impact and release motions as well as no agreed format to store this type of information. For such a reason, a specific structure of an archive, containing several recordings, has been created for ensuring, in particular, reproducibility, searchability, reusability, and human/machine interpretability of the stored data. One of the purposes of this deliverable is indeed to provide public access and description of a first version of such an archive, to allow for testing and further improvement, based on concrete experience in recording impact/release data as well as forecasting the future needs of the entire I.AM. project and beyond.

2.3.1. Data collection

Data will be collected on ad-hoc designed experimental setups, mainly at TU/e for the BOX and TOSS scenario and secondary in other partners local setups for the GRAB scenario. Data collection should have the clear purpose of serving at least one of the four needs of the project (modeling, learning, sensing, control). For this reason, the I.AM. partners agree to share the intention to creating a new archive with the entire consortium ideally prior the initiation of the data collection activity, in order to receive feedback/suggestions on the scenarios and operational conditions to be considered, the information to be recorder, sensors to be used, etc.

During the collection of the data, it will be important to document with detailed descriptions and pictures the setup, including details such as the robot, end-effector, objects, environment. Important information to record also geometric (dimension, shape) and dynamic (mass, inertia, material) properties of the robot(s), object(s), and environment involved in the experiments, in order to allow for the reproduction of the experimental conditions. For commercial products (robots, end-effectors, consumer objects), it is requested to be described the exact model/type of product.

2.3.2. Data structure

The I.AM. **repository** will be built up from several archives. Each **archive** constitutes a set of coherent measurements with a specific purpose. More specifically, each archive is constituted by several **recordings**, containing **datasets**, organized in a **hierarchical structure** (folder/file type structure) and provided with **metadata** (attributes, both at folder and file level) to allow for human and machine search and understanding. The current storage of data makes use of the open-format HDF5, but other formats such as netCDF [1] are also being evaluated as possible alternative.

The proposed structure for each archive is straightforward, being composed by a root folder and several subfolders, each corresponding to a specific recording, and a recording format dataset essentially corresponding to this deliverable section, for making the content of the archive understandable. Visually, the archive content structure can be represented as:

+---<**ARCHIVE'S ROOT**>
+---- <**RECORDING_1**>
+---- <**RECORDING_2**>
.
.
+---- <**RECORDING_N**>
recording_format

Because the number of archives within the I.AM. data repository will grow during the project and after its conclusion with external contributions from other/future projects, it is essential to provide each archive with metadata that are capable of identifying the **purpose** and **type of impact/release motions**, besides the metadata related to time, location, authors, type of object(s), robot(s), environment sensor(s) and associated time-series sensor data. These metadata have the purpose of identify similar archives and, within those, identify specific recordings that share similar features (e.g., impact of a box with a conveyor belt or grabbing of a box via a dual arm robotic system).

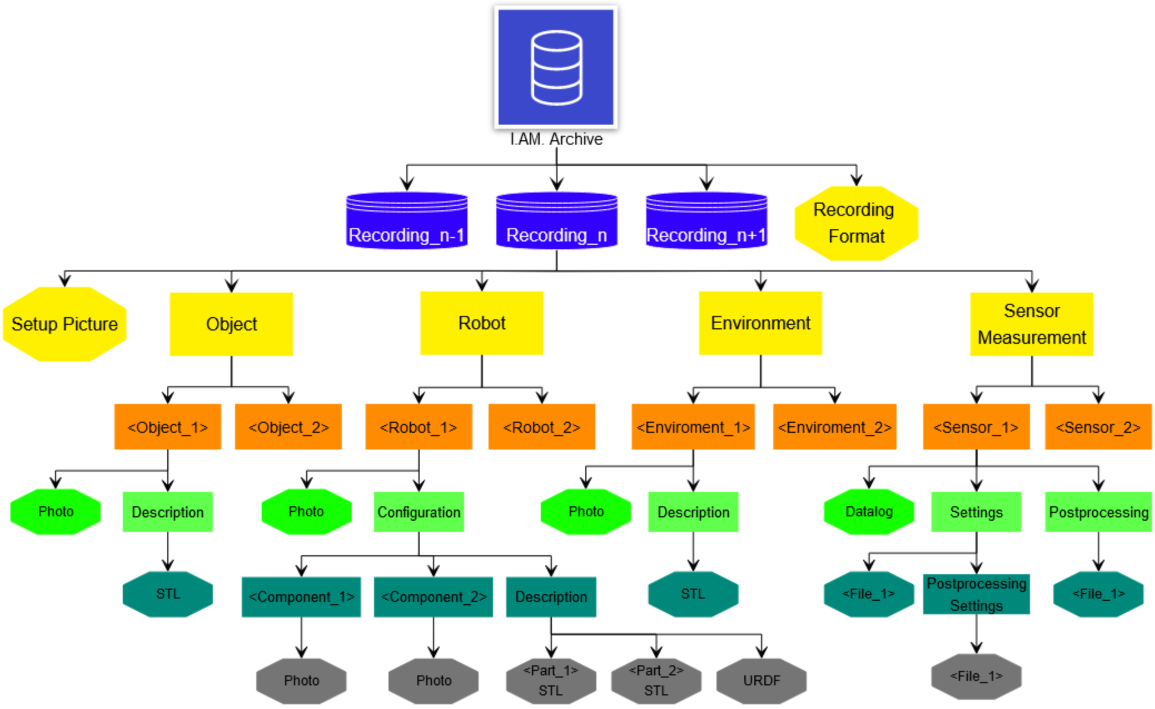

Figure 2: Structure of an I.AM. archive. Each archive contains a set of recordings, as well as a description of the recording forma. Each recording is subdivided into object, robot, environment, and sensor measurement subfolders and contains a picture of the setup used during the data recording. Details in the main text.

For each archive, metadata are stored both at root and recording level, allowing for each archive and each single recording within the archive to be searchable within the entire I.AM. repository. Specific metadata (in particular, the metadata **transition**, related to the contact transition taxonomy detailed in Section 3) have been created to describe different type of impact/release motions. The metadata associated to the archive's root are given in the following list. In this list, possible metadata values are also provided within square bracket for illustration purposes.

ROOT's ARCHIVE metadata

- **description** [box-conveyor impact data for I.AM. project TOSS scenario]
- **keywords** [robotics; non-smooth mechanics; …]
- **project** [H2020 I.AM. EU Project]
- **purpose** [modeling | learning | …]
- **transition [** impact; object; robot | …]
- **robot** [torque-controlled manipulator | collaborative manipulator | …]
- **object** [uniformly filled cardboard box | parcel | …]
- **environment** [conveyor belt | smooth plate | …]
- **sensor** [motion capture; camera; joint encoder | …]
- **author** [<Name Surname 1>; <Name Surname 2>; …]
- **institution** [Eindhoven University of Technology (TU/e) | …]
- **creation timestamp** [<YYYYMMDDThhmmssZ>]

As mentioned in D6.3 Data Management Plan (DMP), **recording naming** should follow the convention

Rec_YYYYMMDDThhmmssZ_TYPE,

providing a natural ordering based on the beginning of the recording (YYYYMMDDThhmmssZ) expressed in Coordinated Universal time (UTC), with T standing for time and Z being standard abbreviation for UTC according to Standard ISO 8061, and *possibly* including an additional markup (TYPE) indicating particular type of recording (useful for visual inspection of the archive by a user). Each recording will be further partitioned in four subfolders (OBJECT, ROBOT, ENVIRONMENT, SENSOR_MEASUREMENT), visually represented as follows:

```
.
.
+---RECORDING_n
|    + setup_photo
|    +---- OBJECT
|    +---- ROBOT
|    +---- ENVIRONMENT
|    +---- SENSOR_ MEASUREMENT
 +---RECORDING_n+1
 .    .
 .
```

The metadata associated to each recording are the following:

RECORDING metadata

- **note** ["box tossed by hand"; "box tossed by robot"]
- **creation timestamp** [<YYYYMMDDThhmmssZ>]
- **recording timestamp** [<YYYYMMDDThhmmssZ>]
- **author** ["Name Surname"]
- **institution** ["Eindhoven University of Technology (TU/e)"; ...]
- **setup location** ["Vanderlande Innovation Lab on TU/e campus"; ...]
- **transition** ["impact; object; environment"]
- **object_environment** ["single; surface; close; stick"]
- **robot_object** ["multi; edge; open"]
- **robot_environment** ["dual; point; partly_closed"]

The last four metadata (transition, object_environment, robot_object, robot_environment) are the metadata used to classify the recording based on the I.AM. taxonomy. Details about the I.AM. taxonomy are provided in Section 3.

The folders OBJECT, ROBOT, ENVIRONMENT, and SENSOR_MEASUREMENT contain information about specific objects, robots, environment, sensors used in that recording, as separate subfolders.

2.3.2.1. The OBJECT subfolder

The subfolder OBJECT will contain, as separate subfolders, the list of all objects and related information whose motion was recorded by at least one sensor in the recording. The folder OBJECT's structure will be thus the following:

```
.
+---OBJECT
|    +--- <OBJECT_NAME_1>
|          + photo
|.         + ... (additional user-defined information)
|    +--- <OBJECT_NAME_2>
|          + photo
|         + ... (additional user-defined information)
.
```

Each object subfolder (<OBJECT_NAME_1>, <OBJECT_NAME_2>, ...) will contain a picture of the object, have possibly additional subfolders providing extra information about the box (material, assembly instruction, ...), and will have the following associated metadata:

<OBJECT_NAME_n> metadata

- **type** [uniformly filled carton box | ...]
- **model** [TUe_BOX_model_1**]**
- **note [** box used for I.AM. TOSS scenario | ...**]**

The folder name **<OBJECT_NAME_n>** allows to give a name to the specific object (such as, e,g, "BOX" or "BOX_1"). The field **type** is the same as the one(s) appearing in root's archive metadata **object.** The root's archive metadata **object** is actually created by collecting all possible entries **type** in the subfolder of the folder OBJECT**,** within all recordings. The field **model** provides information regarding the specific model of box employed (model allows to specify a possibly "user-defined" standard objects). The optional metadatum **note** allows the user to add further descriptions about the object.

2.3.2.2. The ROBOT subfolder

Similar to the folder OBJECT, the folder ROBOT contains a subfolder for each robot present in the recording (e.g., the left and right robotic arms in a dual arm manipulation system). Each robot subfolder contains a subfolder CONFIGURATION to list extra components (e.g., a robot tool changer, a specific end-effector...) that are attached to the robot during the recording as well as specific description files (e.g., URDF + STL files) to allow for the entire robot visualization, as separate subfolders.

```
+---ROBOT
|   +--- <ROBOT_NAME_1>
|   |    + photo
|   |    + CONFIGURATION
|   |      +--- DESCRIPTION
|   |      +--- <COMPONENT_NAME_1>
|   |      |    + photo
|   |      |    + ...
|   |      +--- COMPONENT _NAME_2>
|   |      |    + photo
|   |      |    + ...
|   |       ...
|   +--- <ROBOT_NAME_2>
|        + photo
|.       + CONFIGURATION
|          +--- DESCRIPTION
|          +--- < COMPONENT_NAME_1>
.          .    + photo
```

Similar to the folder OBJECT (and this holds also for ENVIRONMENT folder), each folder ROBOT is provided with the following metadata:

<ROBOT_NAME_n> metadata

- **type** [collaborative manipulator | torque-controlled manipulator | ...]
- **model** [Franka Emika Panda Research | Universal Robots UR10 | ...**]**
- **note [** Robot manipulator used for I.AM. TOSS and BOX scenarios **|...]**

The robot name is stored as name of the folder **<ROBOT_NAME_n>** (e.g., "UR10").

2.3.2.3. The ENVIRONMENT subfolder

A similar structure in terms of subfolder and metadata is given to the ENVIRONMENT subfolder, similar to what done for the subfolders OBJECT and ROBOT. Details are not presented here for sake of brevity, but full description will be given in the final release of the I.AM. repository in D1.4 "Publication of I.AM. dataset (update D1.1)" at M36 (Dec. 2023).

2.3.2.4. The SENSOR_MEASUREMENT subfolder

Lastly, the fourth folder in each recording after OBJECT, ROBOT, and ENVIRONMENT is the folder SENSOR_MEASUREMENT, containing the actual measurements. The word `measurement' has to be understood in a broad sense: marker positions or rigid body pose estimates provided by a motion capture system or rigid link poses obtained from robot joint displacements and robot kinematics, for example, are to be considered as potential measurements appearing in the recording (in place of/in combination with raw infrared camera images or robot joint displacements, respectively). The folder SENSOR_MEASUREMENT has the following structure:

```
+---SENSOR_MEASUREMENT
    +--- <SENSOR_NAME_1 >
    |       + datalog
    |       +--- SETTINGS
    |       |       + <settings_file_1> [ mocap's initialization file | ...]
    |       |       + <settings_file_2> [ mocap's logging file configuration | ...]
    |       |       + ...
    |       |       +--- POSTPROCESSING_SETTINGS
    |       |               + ...
    |       +--- POSTPROCESSING
    .               + ...
    +--- <SENSOR_NAME_2>
    |       + datalog
```

| **+--- SETTINGS**
| **+--- POSTPROCESSING**
.

Each subfolder **<SENSOR_n>** contains the dataset **datalog** (a tabular data where each row corresponds to a specific time instant and each column representing a specific labelled measurement) and two folders: **SETTINGS** and (optionally) **POSTPROCESSING**. The metadata of the dataset **datalog** is defined to fully characterize the properties and data structure of the dataset (frequency, type of data, ...). The folder **SETTINGS** gather all information regarding the initialization and logging of the sensor data and (optionally) extra information in the subfolder **POSTPROCESSING_SETTINGS** about some parts of the setup (=object + robot + environment) that are used to generate the datasets contained in the **POSTPROCESSING** subfolder. The **POSTPROCESSING** subfolder contains processed sensor data that is obtained from the **datalog** and that ease the understanding/visualization of the **datalog** dataset, without the need to run external, possibly time-consuming, routines to generate these postprocessing data (in any case, postprocessing routine will be made available in order to allow for reproducing the postprocessing data). For example, the folder **POSTPROCESSING** contains, for a motion capture system, the rigid transformations with respect to a common frame of reference, shared with the robot manipulator, of various parts of the setup, such as the robot base pose or a falling box pose. In summary, a schematic overview of the archive structure can be seen in Figure 2.

2.3.2.5. Storage of binary files in an I.AM. archive

In various occasions, it is essential to be able to store entire files in an I.AM. archive. In order to do, the following metadata structure is required for a file in the I.AM. archive to be recognized as a binary file

- **file_format** [stl | mp4 | urdf | ...]
- **filename** [<namefile.extension>]
- **note** [STL file of box1 | ...]

2.3.2.6. Metadata associated to a physical component folder in an I.AM. archive

Any folder in an I.AM. archive describing a physical part (such as objects, robots, environment, or parts of objects, robots, environment) must provide at least the following metadata, as already mentioned for objects and robots:

<PART_NAME>

- **type** [<what is this part? Generic description>]
- **model** [<which exact model? Specific description: brand, model, unique identifier>]
- **note [** < further comments that can help understand/reproduce the data/setup>]

2.3.2.7. Naming convention

As a general rule, folder names should start with a capital letter (e.g., Tool_arm). Predetermined folders (e.g., OBJECT, ROBOT) should be also fully in capital letters. Dataset (e.g., datalog), metadata (e.g., transition), and metadata entries (e.g., impact) should be all in small letters, unless part of acronym (e.g., UR10). Use of underscore is encouraged (e.g., robot_arm).

As a final recommendation, it is suggested to have for each folder and file within the I.AM. archive the metadata **note** for provide information about its content.

2.3.3. Data access

Low-level access to the content of each archive of the I.AM. repository will be guaranteed, as mentioned in Section 2.3.2 and D6.3 Data Management Plan, by using a standard open-data format that is provided with I/O routines available in a large number of programming environments (C++, MATLAB, Python, to cite the most relevant to the I.AM. project) as well as a multi-platform generic viewer to inspect the content of the archives (e.g., the HDFview app for HDF5 format). The missing service, to ensure the data is accessible worldwide, is to store the repository and the single archives on a cloud server to allow for remote access.
This web server will contain both a Web interface (HTTP) as well as internet-based data access protocol to enable remote, selective data-retrieval. Example of existing web interfaces related to well-established robotics repository include

- **KIT Whole-Body Human Motion Database** [2]
- **CMU Graphics Lab Motion Capture Database** [3]
- **The KITTI Vision Benchmark Suite** [4]

Regarding the data access protocol, at the time of writing this deliverable, it seems that the open data access protocol (openDAP) [5] could constitute a solution for the current and future need of the I.AM. repository. Another example of data access protocol is the one used in the KIT Whole-Body Human Motion Database, which is the Internet Communications Engine (Ice), an object-oriented middleware framework (more precisely, a remote procedure call) [6] that is however proprietary.

Because setting up such a web server is currently in the making, it has been decided to publish the first archive of the I.AM. project (with format described in Section 2.3.2) in a general-purpose repository server comprising a web interface and a data access protocol. Such a publication, detailed in Section 4 below, has as main purpose that of allowing the consortium partners to access the recorded data, familiarize and give feedback on the current archive format, and experiment with the data access protocol (openDAP), testing selective data download.

The specific option that has been chosen is the "4TU.ResearchData" server [7, 3, 8] which provides long-term (15 years) archive for storing and reusing research data in the technical sciences.

4TU.ResearchData allow the datasets to be stored as netCDF [1] or HDF5 files. The 4TU.ResearchDate service is comparable with the European initiative Zenodo [9], already mentioned and described as one possible options for data storage in D6.3 Data Access Management.

2.4. Data collection plan (when and who)

The time plan for data collection will follow the expected scenarios validation plan, shown in Figure 3. Concisely, the expectations are the following in terms of responsibility for data collection.

Responsible Partner:

- TOSS and BOX scenarios, mainly TU/e, with input from TUM, FRANKA EMIKA, CNRS, and EPFL.
- GRAB scenario: mainly EPFL and CNRS for providing equipment (dual arm KUKA robot, dual Panda arms, sensor I/O logging), support for hardware and acquisition software, performing of experiments; TU/e for setting up experimental conditions and storage of collected data in agreed format with inputs from TUM, CNRS, and EFPL.

Expected recording time:

- First year: TOSS/BOX
- Second year: BOX/GRAB
- Third year: GRAB

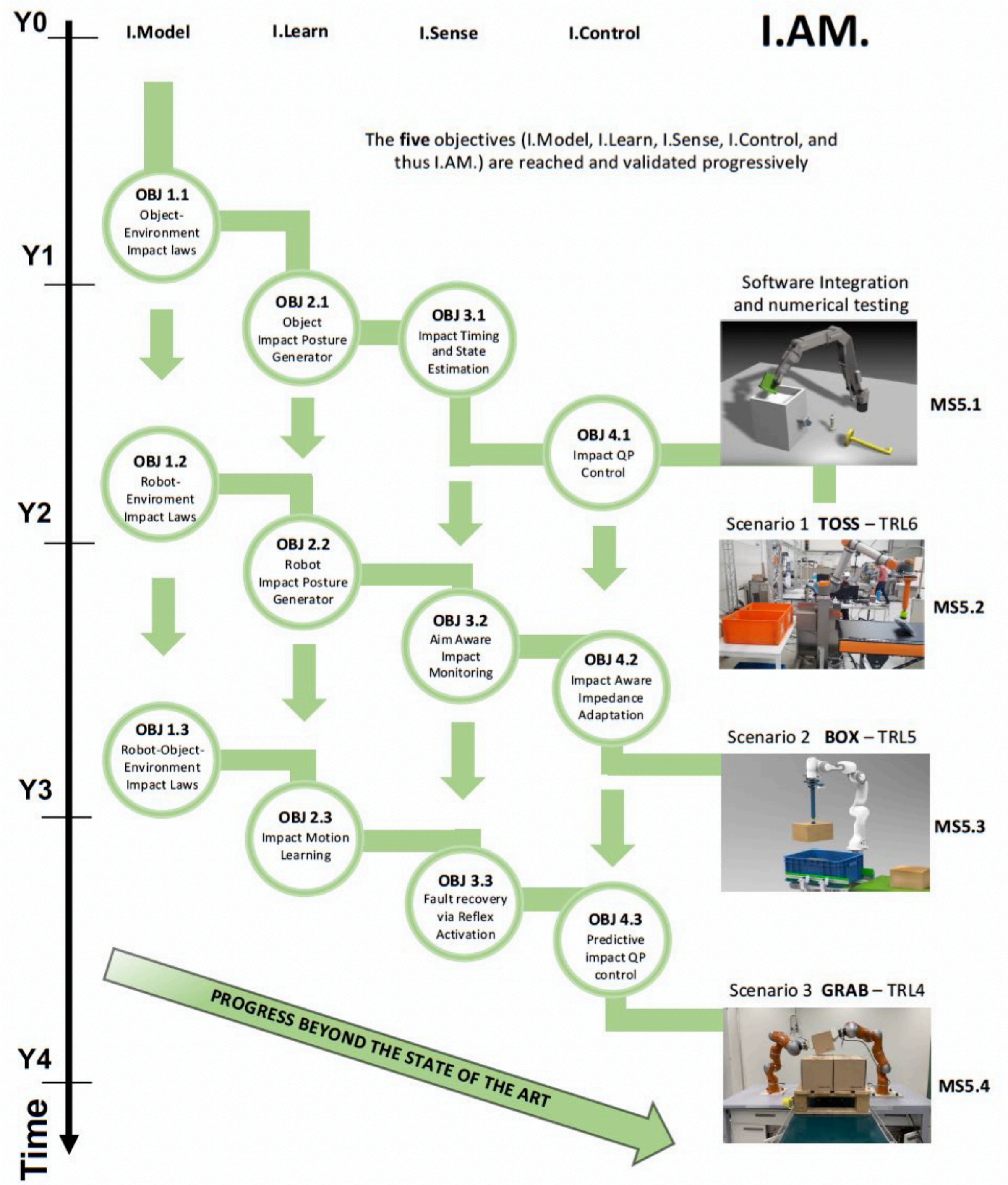

Figure 3: Time plan of the I.AM. project. Data collection plan for different type of impacts and release motions will be aligned with the project time plan.

3. I.AM. CONTACT TRANSITION TAXONOMY

This section provides further details on how time-series of sensor recordings related to a contract transition (impact or release) can be uniquely classified by means of a limited number of metadata and corresponding possible entries. This classification is essential to be able to retrieve specific archives (e.g., just those containing recordings of impact data between an object and a surface or those contain recording of specific impacts between a robot and an object, while in a specific contact state with the environment) out of the (eventually large) I.AM. repository.

As illustration, the classification of impact events related to grabbing a box laying isolated on a table via a dual arm robot can be represented as in Figure 4 which also illustrate the main structure of the taxonomy tree. In the following, the rationale and details of this classification (I.AM. taxonomy) are provided, together with how such information will be stored as metadata in every I.AM. archive.

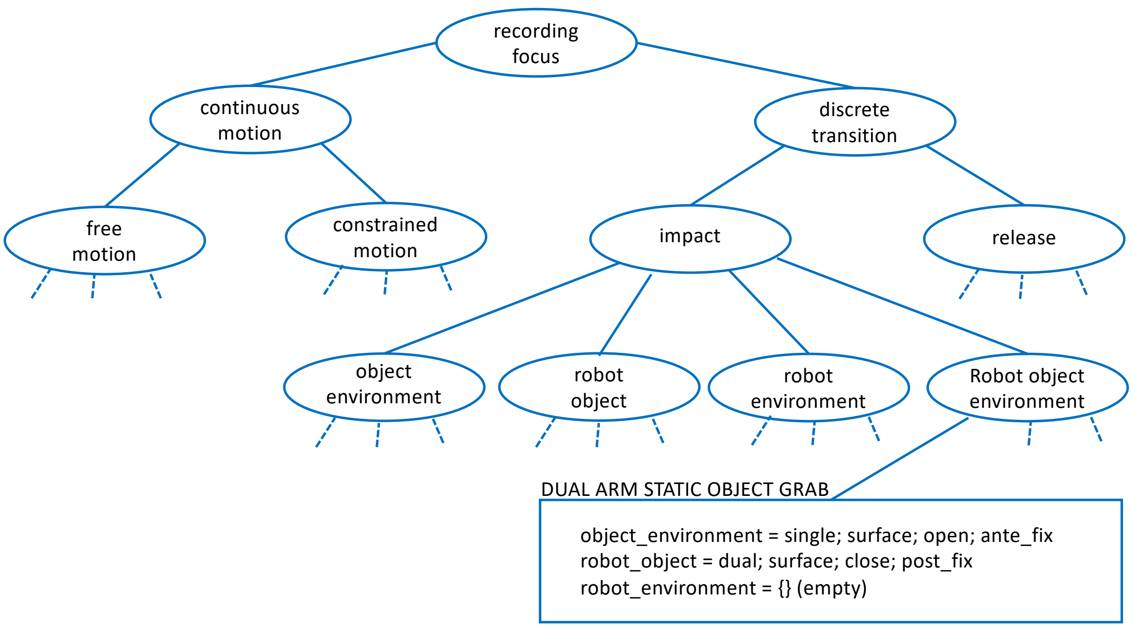

Figure 4: Illustration of I.AM. taxonomy. The drawing illustrates a special leaf in the taxonomy tree associated to a dual arm grabbing of a box, standing isolated on a surface before being picked up. More details are given in the main text.

Manipulation can involve three types of entities simultaneously: the robot(s), the object(s), and the environment. In the I.AM. project, a qualitative metric is proposed based on the type of contact transitions involving these entities. Contact transitions are classified from simpler to more complex, based on four level, depending if the transition involves (i) just the object and the environment; (ii) just the robot and the object; (iii) just the robot and the environment, and (iv) the robot, object, and the environment. Furthermore, the **dynamic contact transitions** are classified either as **impacts** (e.g. a box hitting a table) or **release** (e.g. a box falling off a table). Impacts are considered to be at a higher level of complexity than releases, due to the fact that impacts cause jumps in the velocity of the object, while releases only cause jumps in the acceleration of the object. This preliminary leads to the classify dynamic transitions into a complexity level from 1 to 8, as can be seen in Figure 5.

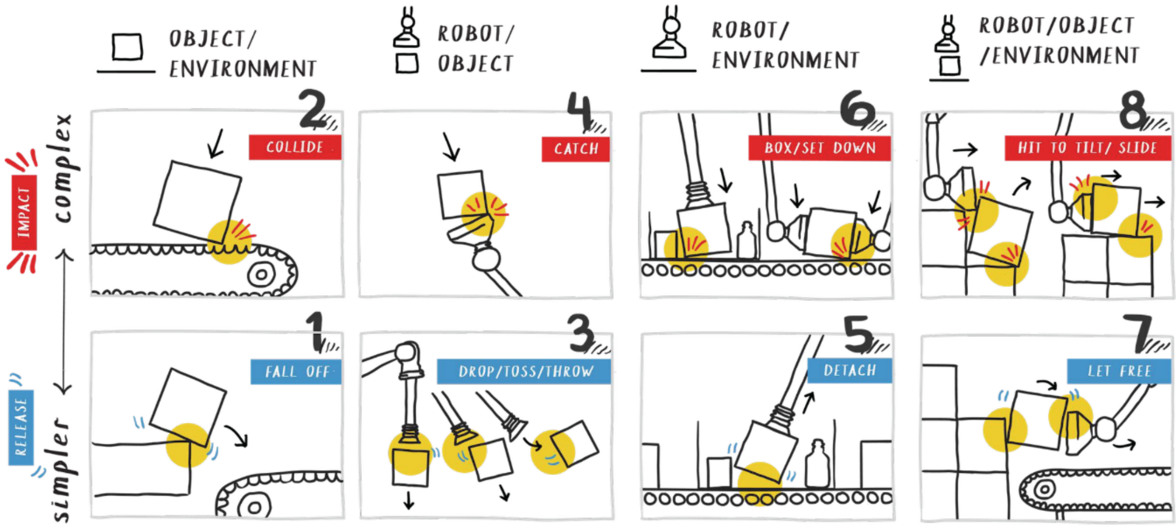

Figure 5: The proposed metric used to cluster dynamic contact transitions in terms of increasing execution.

Correspondingly, each recording of dynamic contact transition in an I.AM. archive can be classified, by means of the metadata

transition

which is a list of the following entries, comprising both the type of transition and the entities involved:

- **impact**
- **release**
- **object**
- **robot**
- **environment**

Only eight accepted combinations for the **transition** entries above are possible, corresponding one-to-one to the eight levels of complexity shown in Figure 1:

1. **release; object; environment**
2. **impact; object; environment**
3. **release; robot; object**
4. **impact; robot; object**
5. **release; robot; environment**
6. **impact; robot; environment**
7. **release; robot; object; environment**
8. **impact; robot; object; environment**

It is essential to understand that impact and release refers to the main driver causing the action: a box grabbed from a table will show both impact between the gripper and the box, but also release between the table and the box. Nonetheless, this is classified simply as impact.

As also explained in Section 2.1, there are situations where it will be important to allow the creation of archives with no contact transitions (e.g., for storing of data to be used to identify the flexible model of a suction gripper). To this end, two additional entries for the metadata **transition** are provided. These two additional entries are

- **free**
- **constrained**

and additional 8 combinations are possible (not listed explicitly here), mimicking what done for the release and impact entries. Just for giving one example, one can use

constrained; object; robot

to describe an archive containing recordings of a robot holding an object.

Besides the metadata **transition**, three other metadata are defined to further detail the content of the archive. These metadata will allow for a finer search within the I.AM. repository (e.g., to get from all recording of impact between an object and the environment, only those showing bouncing on a single point). These three additional metadata are

object_environment
robot_object
robot_environment

For all of these metadata, the following entries are introduced to specify the contact geometry

- **point**
- **edge**
- **surface**

Furthermore, the following additional entries are introduced to specify the number of separate contact location

- **single**
- **dual**
- **multi**

As example, in an impact grasp of a two-finger with a box, the **object_robot** metadata will be

object_robot = dual; point;

To allow for a more detailed level of classification, for each object-environment, object-robot, robot-environment contact pairs one can use the following *position-level* entries

- **bounce** [open before and after the event, contact during the event]
- **open** [contact before the event, no contact after the event]
- **partly_open** [contact before the event, less contact area after the event]
- **stay_closed** [contact before and after the event, same area]
- **further_closed** [contact before the event, more contact area after the event]

- **closed** [no contact before the event and contact after the event]
- **shift** [contact before and after the event, but different areas]
- **partly_shift** [contact before and after the event, overlapping areas]

These eight position-level entries are the enumeration of all possible ante- and post-event combination of the contact interface state between two entities.
Finally, the following ante-event and post-event *velocity-level* entries can be used to describe if relative motion is present at contact interface before and after contact is established:

- **before_move** [relative motion at the contact interface, before the event]
- **before_rest** [no relative motion at the contact interface, before the event]
- **after_motion** [relative motion at the contact interface, after the event]
- **after_rest** [no relative motion at the contact interface, after the event]

The entries **before_move** and **before_rest** cannot be used when the position-level entries **closed** and **bounce** are used, as there was no contact before the event to be described. Similarly, the entries **after_motion** and **after_rest** cannot be used for the position-level entries **open** and **bounce**, as there is not contact after the event to be described. Although it could be possible to further specify the contact motion state, the current level of description is deemed sufficient in this first version of the taxonomy.

As first illustrative example, an impact corresponding to a dual arm robot grabbing a box that was sliding, isolated, on a surface can be described with the following metadata

1. **transition = impact; object; robot; environment**
2. **object_environment = surface; single; open; before_move**
3. **robot_object = surface; dual; closed; after_rest**
4. **robot_environment =** { } (empty)

The metadata above corresponds to a possible contact transition in the GRAB scenario. The metadata describe a scenario where the end effectors stick to the side of the box after impact.

As second illustrative example, the release motion corresponding to suction gripper letting a box fall, corresponding to a possible contact transition in the TOSS scenario, is described as

1. **transition = release; object; robot**
2. **object_environment =** { } (empty)
3. **robot_object_ = surface; single; open; before_rest**
4. **robot_environment_ =** { } (empty)

4. FIRST PUBLISHED VERSION OF I.AM. ARCHIVE

This section will describe the first published version of I.AM. archive [10], as example of which type of archive files will be stored in the I.AM. repository. As mentioned earlier in Section 2, the archive is the result of experiments executed related to the scenario of TOSS. First an overview is given of the archive, including a photo of the setup. Then, we will consider one specific recording and demonstrate the data structure as discussed in Section 2 of OBJECT, ROBOT, ENVIRONMENT, and SENSOR_MEASUREMENT.

4.1. Archive overview

The archive under consideration contains recordings of experiments that are executed under the scenario of TOSS. In these recordings, a UR10 robot is used to toss boxes on a conveyor belt. The purpose of these experiments to validate a modeling framework. This modeling framework is used within the I.AM. project to predict the end pose of a certain box on a conveyor belt, after it is tossed. This means that the involved contact transitions are between the object and the environment, which in these recordings is a box and a conveyor. All the recordings in the archive are executed by Luuk Poort at the Innovation Lab of Vanderlande, located at the TU/e. To summarize, the metadata of this archive, as discussed in Section 2.3.2, is then given as follows.

- **description =** box-conveyor impact data for I.AM. project TOSS scenario
- **keywords =** robotics; nonsmooth mechanics
- **project** = H2020 I.AM. EU project
- **purpose =** modeling
- **transition =** impact; object; environment
- **robot =** collaborative manipulator
- **object** = uniformly filled cardboard box
- **environment** = conveyor belt; multiplex smooth surface
- **sensor** = motion capture; camera; joint encoder
- **author** = Luuk Poort; Maarten Jongeneel
- **institution** = Vanderlande Innovation Lab at TU/e campus
- **creation timestamp** = 20200929T121000Z

As discussed in Section 2, each recording has the same hierarchical structure. In Figure 6, this hierarchical structure of the archive is given for one specific recording. Each recording contains the folders OBJECT, ROBOT, ENVIRONMENT, and SENSOR_MEASUREMENT. Files (HDF5 dataset) are indicated by octagons while folders (HDF5 groups) are indicated by rectangles. Several folders content is not explicitly shown (indicated by the folder icon with a plus sign) for the sake of brevity. The figure does not include, for each folder and file, the metadata, that are instead described in the main text below, following the structure detailed in Section 2.3.2.

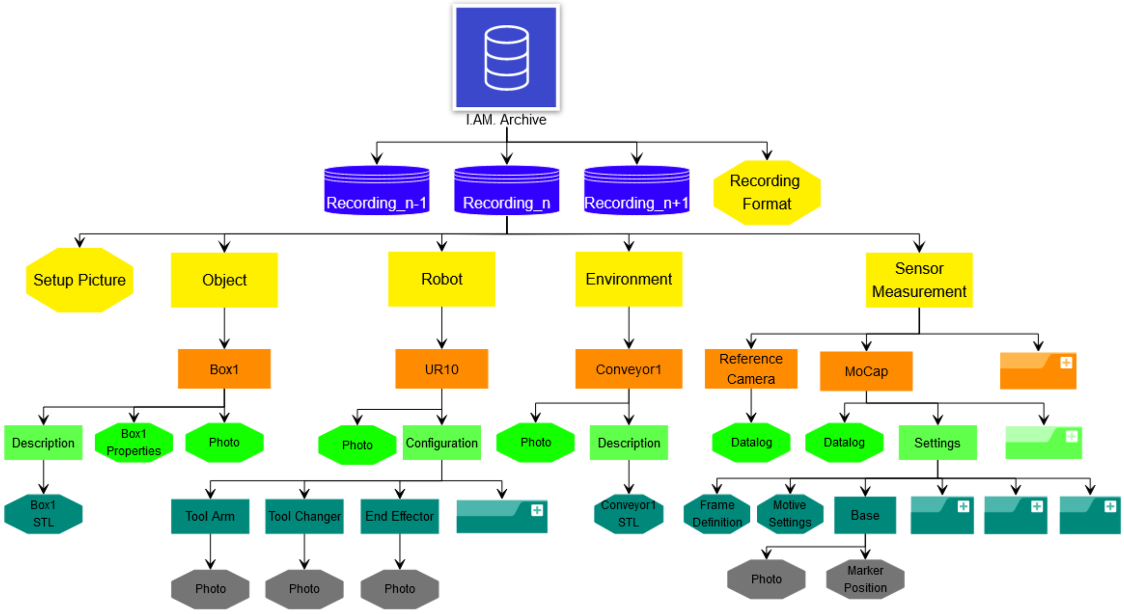

Figure 6: Structure of the I.AM. archive for the box-conveyor impact recording described in this section.

In the following subsections, the hierarchical structure of a single recording is further elaborated. The specific recording under consideration is a recording of an experiment executed at the Vanderlande's Innovation Lab located on TU/e campus. In this experiment, a carton box is tossed by the UR10 robot on a moving conveyor belt. This means that the transition under consideration is an impact of the object with the environment, where the focus is recording (corner) impacts between the box and the conveyor surface. Additionally, neither interaction between the robot and the environment nor the robot and the object is considered. Given these settings, the metadata of a representative recording in this archive is given as follows:

<RECORDING_n> metadata

- **note** = Box-conveyor impact. Box tossed by robot
- **creation timestamp =** 20200626T085406Z
- **recording timestamp =** 20200527T142459Z
- **setup location** = Vanderlande Innovation Lab at TU/e campus
- **author** = Luuk Poort
- **institution** = Eindhoven University of Technology (TU/e)
- **transition =** impact; object; environment
- **object_environment** = point; single; closed; after_motion
- **robot_environment =** {} (empty)
- **robot_object =** {} (empty)

Note that the actual recording data will contain more than just one single impact for toss. Indeed, each recording contains the whole motion of the robot from tossing to the rest moment of the box on the conveyor. However, the taxonomy metadata (transition, object_environment, robot_environment, robot_object) is used to describe what the data is intended to be used for: in this specific case, store box-conveyor corner impacts for impact model identification.

In the main folder of the recording the setup picture is stored. This picture gives an overview of the experimental setup and is given in Figure 7: Picture of the experimental setup. The different components are indicated for reference.Figure 7.

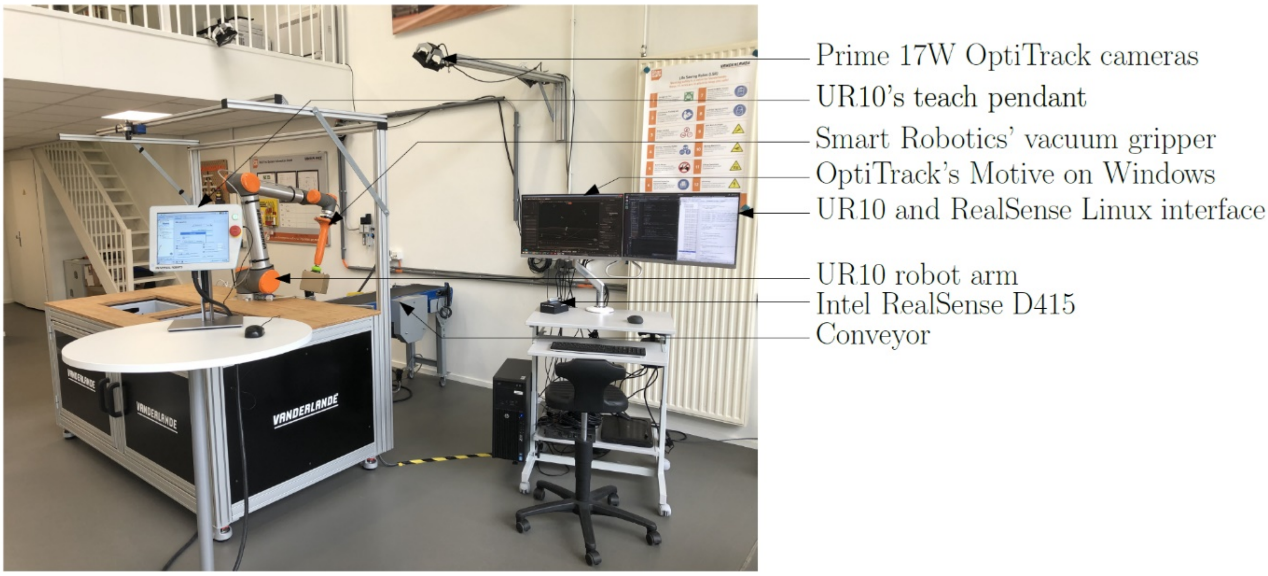

Figure 7: Picture of the experimental setup. The different components are indicated for reference.

4.1.1. the OBJECT subfolder

In the specific recording under consideration, a carton box is used, filled with sculpture block foam. The name given to the box is simply box1. This makes that the OBJECT subfolder contains one subfolder, named after the only object which is used in the experiment. The metadata of this folder is then given as follows:

Box1 metadata**:**

- **type =** uniformly filled cardboard box
- **model =** TUe_box_model_1
- **note** = carton box filled with sculpture block foam

Considering the structure given in Section 2, the folder Box1 should contain a photo of the object, which is shown in Figure 8.

Additional information (user-defined dataset) is also included. In particular, the dimensions (LxWxH) and mass of the box are measured, and their values are stored in the folder Box1. Based on the -dimensions of the box and the marker positions, a 3D CAD drawing is created of the box. The moments of inertia with respect to the center of mass of the box are computed based on this CAD drawing and stored in the Box1 folder. These values are also summarized in Table 1, where the directions x, y, and z correspond to the length, width, and height, respectively. Additionally, Table 1 shows the properties of the inner and outer material of box1. In the archive, all the data of Table 1 is stored in a single file **box1_properties**.

Note that the positions of the markers on top of the box, that used for the motion capture system, are not part of the description of the model box and thus not stored in folder Box1. Rather, they are stored in the SENSOR_MEASUREMENT folder as they are related to a particular measurement, as it will be discussed in a later subsection.

Table 1: Details of the TUe_BOX_model_1 model (box model used for tossing recordings)

Parameter	TUe_BOX_model_1	Unit
Dimensions (L x W x H)	0.117 x 0.091 x 0.066	m
Mass	0.088	kg
Moments of inertia w.r.t. CoM (I_{xx}, I_{yy}, I_{zz})	$(1.04 \cdot 10^{-4}, 1.54 \cdot 10^{-4}, 1.82 \cdot 10^{-4})$	kg m^2
Inner material	Sculpture Block SB30. Density of 100 kg/m^3 [11]. Manually cut to fit size of box1.	-
Outer material	Carton	-

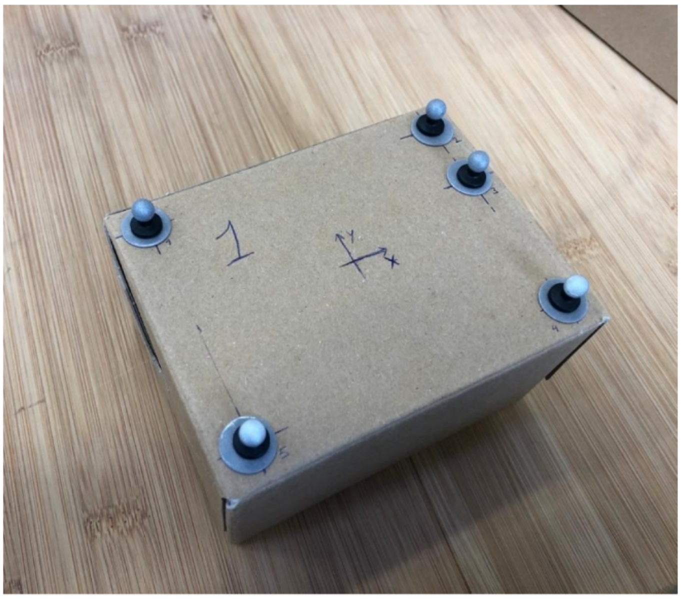

Figure 8: Photo of box1.

The folder Box1 contains a subfolder, named DESCRIPTION. In this subfolder, an STL file of box1 is stored as binary file, named **box1_geometry**. Such a file is stored binary in the archive, with the following metadata (as described in Section 2.3.2.5):

box1_geometry metadata**:**

- **file_format** = stl
- **filename =** box1_geometry.stl
- **note** = STL file of box1

Given this metadata, the user will be able to write the binary file to the correct file extension.

In summary, the OBJECT folder will thus have the following structure:

```
+---OBJECT
|    +--- Box1
|          + photo
|          + box1_properties
|          + ---DESCRIPTION
|                + box1_geometry
```

4.1.2. the ROBOT subfolder

Starting from the general structure laid out in Section 2.3.2, in this section the specific robot configuration used in the recording is explained. The structure of the ROBOT folder is the following:

```
+---ROBOT
|    +--- UR10
|    |    + photo
|    |    + ---CONFIGURATION
|    |          +--- DESCRIPTION
|    |          +--- End_effector
|    |          |    + photo
|    |          +--- Tool_arm
|    |          |    + photo
|    |          +--- Tool_changer
|    |          |    + photo
```

The robot name used is UR10. This subfolder contains a photo of the configuration, shown in Figure 9 and the subfolder called CONFIGURATION.

Figure 9: Photo of the robot configuration, showing the robot arm (UR10), the custom suction gripper and the tool changer

The UR10 metadata is

UR10 metadata**:**

- **type =** collaborative manipulator
- **model** = Universal Robots UR10
- **note** = robot manipulator used for I.AM. TOSS scenario

The folder CONFIGURATION contains further information about the specific robot configuration used. This includes the tool changer, tool arm, and end effector. Each of these parts are given a subfolder under CONFIGURATION, together with a subfolder DESCRIPTION which store geometric/inertial information for the entire robot. Hence, the structure of CONFIGURATION, reported here for reader ease, is given by

+---**CONFIGURATION**
| +--- DESCRIPTION
| +--- End_effector
| +--- Tool_arm
| +--- Tool_changer

In the following subsections we will further elaborate the content and metadata of these subfolders.

4.1.2.1. The DESCRIPTION subfolder

The subfolder DESCRIPTION contains all the STL files of all the different parts of the configuration and a URDF file. All of these files are stored as binary files according using the metadata described in Section 2.3.2.5:

- **file_format**

- **filename**
- **note**

The field **filename** is very important for the STL files, as these exact names are referred to in the URDF file. Given the parts of the configuration, the DESCRIPTION folder has the following structure:

+---**DESCRIPTION**
| + **base** (STL file)
| + **end_effector** (STL file)
| + **forearm** (STL file)
| + **shoulder** (STL file)
| + **tool_arm** (STL file)
| + **tool_changer** (STL file)
| + **upper_arm** (STL file)
| + **wrist_1** (STL file)
| + **wrist_2** (STL file)
| + **wrist_3** (STL file)
| + **urdf** (URDF file)

The DESCRIPTION folder has as one metadata

note = folder containing the STL files and URDF of the UR10

in accordance with what suggested in Section 2.3.2.

4.1.2.2. Tool changer

The tool changer is a physical part added to the UR10. Therefore, this folder has the metadata associated to a part, as discussed in Section 2.3.2, which is given as

Tool_changer metadata**:**

- **type** = custom tool changer
- **model** = SIR robot flange
- **note** = composition of FIPA SR90-A-SL tool changer and flange complying with Standard ISO 9409-1-50-4-M6

As described in the note, the tool changer is in fact an assembly, developed by Smart Robotics, one of I.AM. project's partner. The main part of this assembly is the tool changer by FIPA, the SR90-A-SL. As this tool changer cannot be mounted directly to the robot, a flange is added that complies with the mounting of the UR10 (ISO 9409-1-50-4-M6). Additional parts are added that allow the flow and control of compressed air for a suction cup, which will be further discussed in Section 4.1.2.4. Furthermore, a photo is added of the tool changer, which is shown in Figure 10.

Figure 10: Photo of the tool changer, mounted between the tool arm and the UR10.

4.1.2.3. Tool arm

The tool arm is also a physical part added to the UR10. Therefore, this folder has the metadata associated to a part, as discussed in Section 2.3.2, which is given as

Tool_arm metadata:

- **type** = custom tool arm
- **model** = GS02-v05
- **note** = Smart Robotics custom tool arm

The data stored in the user-defined folder Tool_arm is a photo, which is shown in Figure 11. Note that the markers used for the mocap system are also visible in the picture (marker positions are stored in the SENSOR_MEASUREMENTS subfolders, as for the box's).

Figure 11: Photo of the tool arm, located between the tool changer and the end effector.

4.1.2.4. End effector

The end effector is also a physical part added to the UR10. Therefore, this folder has the metadata associated to a part, as discussed in Section 2.3.2, which is given as

End_effector metadata:

- **type** = Piab piGRIP suction cup
- **model** = G.FLI70S.B3.S1.G38M.00
- **note** = suction cup gripper

As the metadata already shows, the end effector used in the recording is a bellows suction cup from Piab. The specific configuration of the suction cup consists out of a number of parts: (1) a Soft FLI70 lip type (FLI70S); (2) three bellows (B3); (3) a S1 support (S1); and (4) a male G-thread 3/8" fitting (G38M); [12]. This information is also stored in the **model** metadata, which complies with the naming configuration of Piab. This suction cup is chosen in order to deal with the large number of different packages. The bellows in combination with the soft foam on the lip make sure that the suction cup can handle the compliance and large dimensional tolerances, which makes it suitable for handling the large versatility in packages. The data stored in the End_effector folder is a picture, which is shown in Figure 12. Markers used for the mocap system are also visible in the picture.

Figure 12: Photo of the end effector, mounted to the tool arm.

4.1.3. The ENVIRONMENT subfolder

The environment considered in the specific recording is a small conveyor belt. The name given to this conveyor is conveyor1, which therefore also is the name of the subfolder in the ENVIRONMENT folder. Such a folder will have for the box-conveyor recording example treated in this section the following structure (within parenthesis, further explanations provided here for clarity):

+---**ENVIRONMENT**
| **+--- Conveyor1**
| **+ photo** (image of the conveyor)
| **+ --- DESCRIPTION**
| | **+ conveyor1_model (**STL file**)**

As for every folder describing a physical part in an I.AM. archive, the metadata associated to the conveyor1 folder are

Conveyor1 metadata

- **type** = conveyor belt
- **model =** custom Vanderlande small belt floor conveyor
- **note =** conveyor consisting of a rubber belt sliding over a smooth metal surface

The folder Conveyor1 contains a photo of the conveyor, which is shown in Figure 13.

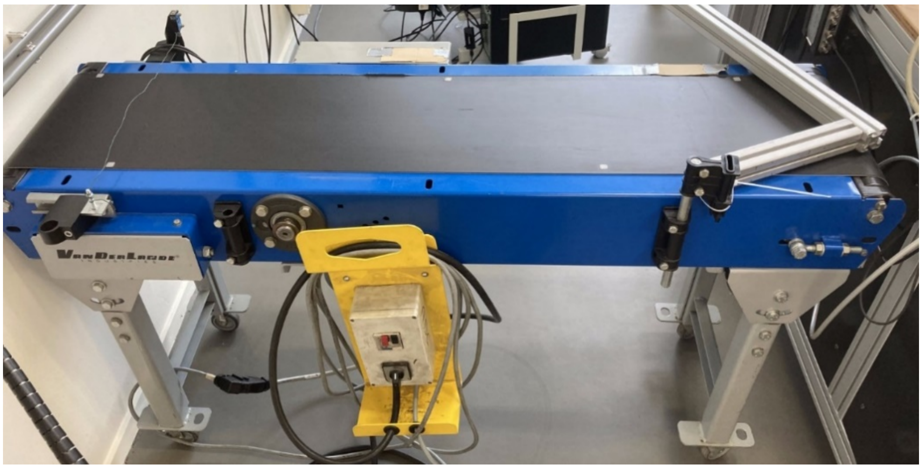

Figure 13: Picture of the conveyor belt used in the recording of the tossing experiments and stored in the archive

Besides a photo, the folder Conveyor1 contains the DESCRIPTION subfolder. In this subfolder, an STL file of the conveyor is stored in binary format, named conveyor1_geometry. In order to allow the reader to interpret the file correctly, the following metadata is added:

conveyor1_geometry metadata:

- **file_format =** stl
- **filename** = conveyor1_geometry.stl
- **note =** STL file describing the geometry of the conveyor1

As described in Section 2.3.2, binary files stored in an I.AM. archive have the metadata **file_format** and **filename** metadata in order to allow to interpret/extract the original file when necessary.

4.1.4. The SENSOR_MEASUREMENT subfolder

The sensors used in the recording are (1) an OptiTrack motion capture system (abbreviated as Mocap), (2) an Intel RealSense D415 used as reference camera, and (3) data logging from the UR10 sensors, containing both encoder data as well as trigger signals. Each sensor has its own subfolder in the SENSOR_MEASUREMENT folder, such that its structure is given as

+---SENSOR_MEASUREMENT
| +--- Mocap
| +--- Reference_camera
| +--- UR10_sensors

The individual sensors and their measurement data are discussed in further detail in the sections below.

4.1.4.1. Content of the **Mocap** folder

An OptiTrack motion capture system is used to record the various experiments. The setup, visible in Figure 7, contains four Prime 17W OptiTrack cameras [13] and an eSync and PoE switch [14]. This allows to record the experiments at a frequency of 360Hz, with sub-millimeter accuracy. Tracking of objects is achieved by the use of markers, placed on the objects to be tracked. The Prime 17W cameras transmit IR light, which reflects on the markers back to the cameras. Making use of multiple cameras, the position of a marker in 3D can be tracked. This makes that the metadata of the **Mocap** folder is given as

Mocap metadata:

- **type** = Motion Capture System
- **model =** OptiTrack Prime 17W
- **note** = OptiTrack mocap made of four Prime 17W cameras and an eSync-and-PoE switch

The OptiTrack mocap comes with a proprietary software program called Motive. In Motive, marker positions can be tracked. These data can be then exported as a comma separated value (.csv) file at the end of the recording. Besides the marker positions, rigid bodies can be associated to a group of markers. The pose of these rigid bodies is represented by a body fixed frame, which can be assigned by the user with respect to the marker positions. The body fixed frames of each rigid body reconstructed by the software Motive can then be written to the .csv as well, by means of its position (x,y,z) and orientation as unit quaternion (q0,q1,q2,q3) with respect to the user-defined world frame recognized in Motive. In the recording under consideration, the following rigid bodies are tracked:

- base of the robot
- tool arm of the robot
- end effector of the robot
- box 1

As a result, for each of these rigid bodies the position, orientation, and individual marker positions are stored in the recording **datalog**. The conveyor1 is tracked by means of 18 individual markers, as only a subset of all its markers can be viewed at a time. Each of the markers have an (x,y,z) coordinates, which is stored in **datalog**.

Besides the **datalog**, the **Mocap** folder contains the subfolders **SETTINGS** and **POSTPROCESSING**, such that the **Mocap** folder has the following structure:

+---**Mocap**
| + datalog
| +--- POSTPROCESSING
| +--- SETTINGS

The SETTINGS folder contains all the settings that were used to create the **datalog**. This includes the markers positions for each rigid body stored in subfolders, a Motive settings file called **motive_setting** and a schematic drawing called **frame_definition,** indicating the reference frames of each rigid body. This means the SETTINGS folder has the following structure:

+---**SETTINGS**
| + motive_settings
| + frame_definition
| +--- Base
| +--- Box1
| +--- End_effector
| +--- Tool_arm

Each of the rigid bodies tracked in motive have physical markers placed on them. Therefore, each folder under SETTINGS referring to a rigid body contains a table **marker_position**, where these marker positions are stored and a picture illustrating the marker positions with respect to a body-fixed frame. These marker positions are measured by hand, but then refined in Motive. To illustrate this concept in this document, we give as example that of the **Base**, which in fact contains information about a tool placed around the base of the robot to track its position. The subfolder **Base** has the following structure:

+---**Base**
| **+ photo**
| **+ marker_position**

These marker positions are 3x1 arrays expressing the (x,y,z) coordinate of the center of the marker with respect to the body fixed frame. For the base, these values are given in Table 2.

Table 2: Marker positions for Base

Marker	Position (x,y,z)	Unit
1	$(-0.175, -0.000596, 0.008)$	m
2	$(0.132, -0.0905, 0.008)$	m
3	$(0.130, 0.0908, 0.008)$	m
4	$(0.152, 0.0456, 0.008)$	m

This frame is illustrated in the picture and stored as photo, which for the base can be seen in Figure 14.

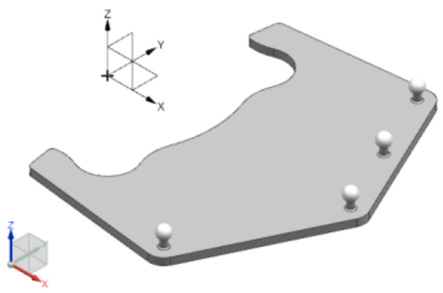

Figure 14: Schematic picture of the base tool with markers and body fixed frame visible.

Similar data is stored in the rigid bodies Box1, End_effector, and Tool_arm, but this is not shown here for the sake of brevity.

The file **frame_definition**, contained in the folder **SETTINGS**, provides an illustration of the setup, where all the body-fixed frames used in the mocap recording are visible, including the Motive world frame.

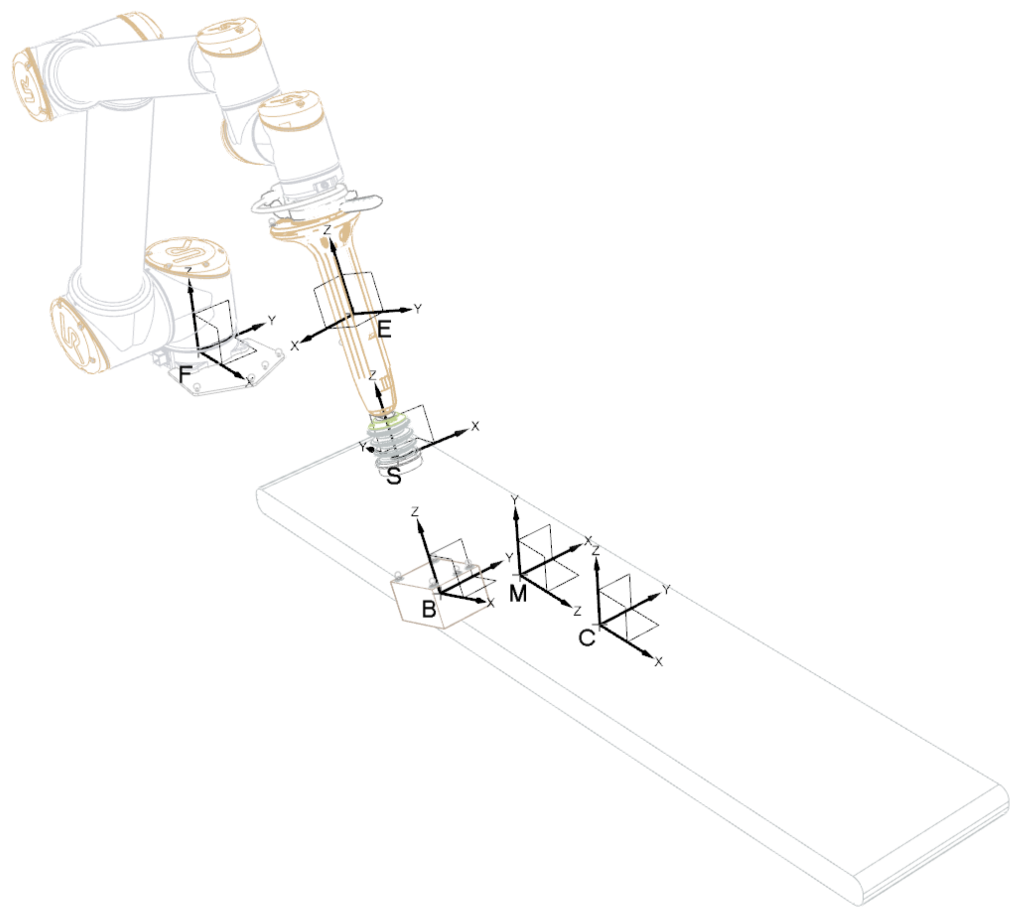

*Figure 15: Schematic picture **frame_definition** containing all the frames of the rigid bodies, and the mocap world frame M.*

The **frame_definition** picture is shown in Figure 15, where the Motive world frame is indicated by the letter M. The frame M is defined for each recording (the user positions it manually before each recording) and all markers positions and body frames stored in the archive **datalog** are expressed with respect to this frame.

The rigid bodies and recording settings are stored in **motive_settings**, which is a user settings file that can be imported into Motive and stored in the I.AM. archive for reproducibility of results.

The **POSTPROCESSING** folder contains the postprocessed data. This folder contains the subfolders **Box1**, **Conveyor1**, **End_effector**, and **Tool_arm**. In each of these subfolders, time-indexed 4x4 transformation matrices are stored, describing the pose of the object with respect to the frame of the base, which in Figure 15 is illustrated by frame F. The **POSTPROCESSING** data allows to quickly compare different recordings as all rigid body poses are referred to the same frame F, without the need of executing any processing of the **datalog** dataset. For the conveyor, an additional dataset velocity is added, which contains the velocity vector of the conveyor belt, also expressed in terms of frame F.

4.1.4.2. Content of the **Reference_camera** folder

Next to a motion capture system, a RGBD camera is used to record the experiments. The recordings with this camera allow to quickly review the experiment, without having to go through the mocap data. The camera records at a quality of 640x480 pixels in RGB at 30Hz. Although the RGBD camera used allows for recording of depth images, only RGB data is stored. The metadata associated to the folder **Reference_camera** is therefore given as

Reference_camera metadata**:**

- **type =** USB-powered depth camera
- **model** = Intel RealSense D415
- **note** = USB-powered depth camera. Consists of a pair of depth sensors, RGB sensor, and infrared projector. Only RGB data is stored

The actual video coming from the camera is stored in the **Reference_camera** folder as a binary file named **datalog**, containing the metadata

datalog metadata:

- **file_format** = mp4
- **filename =** reference_video.mp4
- **note** = reference video footage of the experiment.
- **resolution** = 640x480
- **duration** = 129 frames
- **frame_rate** = 30 fps

4.1.4.3. Content of the **UR10_sensor** folder

The UR10 robot is able to output various types of information. The total list of possible outputs can be found on the website of Universal Robots, in the Real Time Data Exchange Guide [15]. The choice is made to store the actual joint positions and actual TCP pose, together with two additional signal called *trigger* and *gripper*. A trigger is sent to the mocap system to start and stop the recording, which will be discussed in the next section, and this trigger signal is stored in the datalog. The use of the gripper is logged as well, which refers to the specific Piab suction gripper to be turned on or off. All this data is stored in the **datalog**, under the **UR10_sensor** folder. The **UR10_sensor** folder will have the following structure:

+---**UR10_sensor**
| **+ datalog**
| **+--- POSTPROCESSING**

The metadata associated to the "UR10" folder is then given as

UR10_sensor metadata:

- **type** = UR10 sensors (joint positions, actual TCP pose, trigger, gripper on/off)
- **model =** UR10 embedded sensors
- **note =** selection of UR10 sensor reading, important for the tossing experiment

Next to the **datalog**, a **POSTPROCESSING** folder exists. Similar to what is done with the mocap POSTPROCESSING folder, this folder contains time-indexed 4x4 transformation matrices for each part of the robot (links, end effector), expressing its pose with respect to the base frame F. Once again, this postprocessing data makes comparison of different recording easier, without the need to run any conversion script operating on the original **datalog** data. All information for performing such postprocessing is however provided in the archive.

4.1.5. Communication

In this section, additional information is given about the communication of the different components during the recording. This information is not required to understand what the data means, but rather to understand how it was collected and thus allowing for reproducibility of the experimental data. The system components that communicate information are the robot arm with a gripper, the motion capture system, and the video camera. The object (**box1**) does not communicate to the other components and is only tracked by the motion capture system. The environment (**conveyor1**) is also operated in open loop and communicates its position and velocity to the motion capture system by markers on the belt. The remaining components require control and/or evaluation of their states. These interactions are visualized in Figure 16. This information (description and images) is stored as PDF document in the I.AM. archive's root to allow for completeness and allowing reproducibility of the experiment.

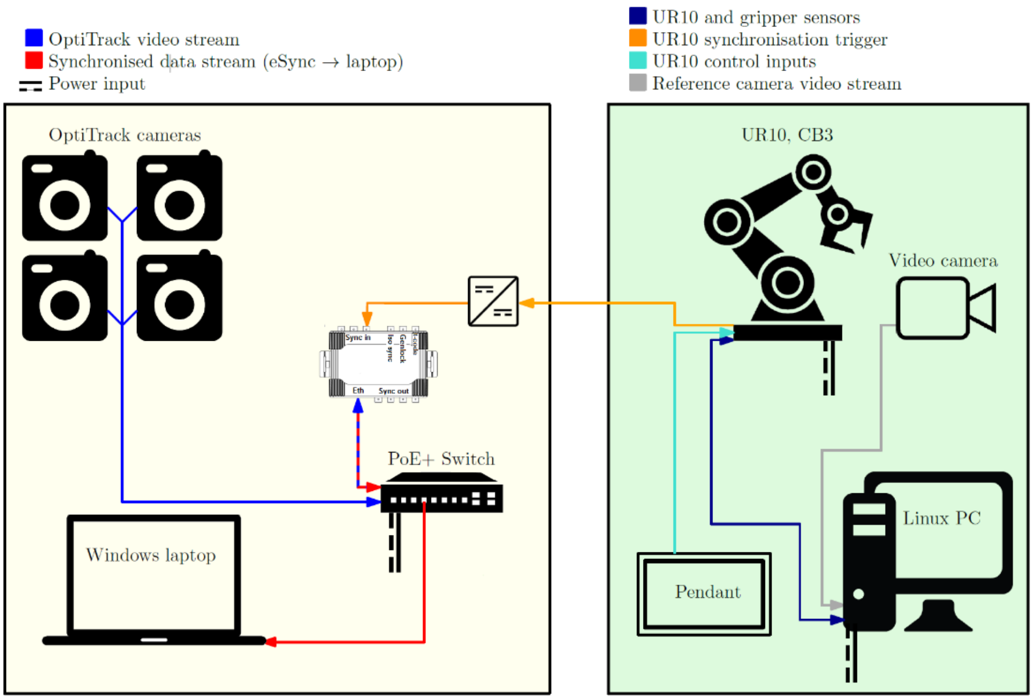

Figure 16: Interaction between the different components of a recording.

At the beginning of the recording, the reference camera is started. The robot is holding the object and is waiting for a signal of the Linux PC. This signal is sent when the first frame of the reference camera is captured. The robot then executes its pre-defined program and sends a trigger signal to the motion capture system to start recording. While the reference camera and robot log their data for a fixed time period, the motion capture system waits for trigger signals from the robot. Also, conveyor1 is triggered by the robot. In summary, the trigger signals of the experiment are graphically depicted in Figure 17.

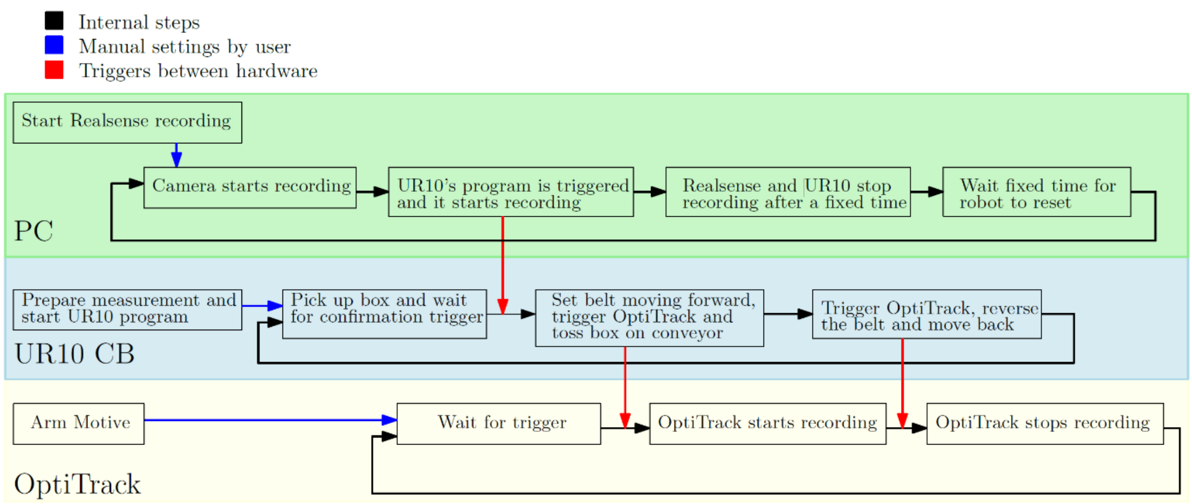

Figure 17: Diagram of the dependencies within the setup's software.

5. CONCLUSION

This deliverable D1.1 provides a first detailed description of the I.AM. repository structure, including the I.AM. taxonomy. The I.AM. taxonomy will be used to classifying impact and release motions (dynamic contact transitions) and will allow for effective search of recordings containing specific motions within the entire repository. The current version of the I.AM. taxonomy is extremely rich, allowing to differentiate among at least 8 x 8 x 8 x 8 = 4096 different impact or release motions. The project will continue till M48 (December 2024) and this deliverable will be updated in M36 (December 2023), outlining the recordings for the TOSS, BOX, and GRAB scenarios, an updated version of the I.AM. taxonomy and I.AM. repository structure, as well as the implemented I.AM. repository's web interface and data access protocol to allow for open access of the dynamic contact transition recordings.

6. REFERENCES

[1] "netCDF," [Online]. Available: https://www.unidata.ucar.edu/software/netcdf/docs/. [Accessed September 2020].

[2] "KIT Whole-Body Human Motion Database," [Online]. Available: https://motion-database.humanoids.kit.edu/. [Accessed September 2020].

[3] "CMU Graphics Lab Motion Capture Database," [Online]. Available: http://mocap.cs.cmu.edu/. [Accessed September 2020].

[4] "The KITTI Vision Benchmark Suite," [Online]. Available: http://www.cvlibs.net/datasets/kitti/raw_data.php. [Accessed September 2020].

[5] "OPeNDAP protocol," [Online]. Available: https://www.opendap.org/. [Accessed September 2020].

[6] "Ice -- Comprehensive RPC Framework," [Online]. Available: https://zeroc.com/products/ice. [Accessed Sep 2020].

[7] "4TU.ResearchData," [Online]. Available: https://data.4tu.nl/. [Accessed September 2020].

[8] "Creative Commons Licenses," [Online]. Available: https://creativecommons.org/licenses/?lang=nl. [Accessed September 2020].

[9] "Zenodo," [Online]. Available: https://zenodo.org/. [Accessed 2020].

[10] L. Poort, M. Jongeneel and A. Saccon, *First version of an I.AM. archive, dataset, DOI: 10.4121/13007474*, Eindhoven: 4TU.ResearchData, 2020.

[11] "Sculpture Block 30 series," 29 September 2020. [Online]. Available: http://www.sculptureblock.eu/en/sculpture-block-collection/sculpture-block-30-series/.

[12] "piGRIP suction cup," Piab, [Online]. Available: https://www.piab.com/about/innovations/pigrip--the-configurable-suction-cup/. [Accessed July 2020].

[13] "OptiTrack Prime 17w," OptiTrack, [Online]. Available: https://optitrack.com/products/prime-17w/. [Accessed July 2020].

[14] "OptiTrack eSync," OptiTrack, [Online]. Available: https://optitrack.com/products/esync-2/. [Accessed July 2020].

[15] "Universal Robots RTDE Guide," Universal Robots, [Online]. Available: https://www.universal-robots.com/articles/ur/real-time-data-exchange-rtde-guide/. [Accessed Sep 2020].

[16] "H5serv," HDF Group, [Online]. Available: https://h5serv.readthedocs.io/en/latest/DatasetOps/index.html. [Accessed July 2020].

[17] "Creative Commons Zero," [Online]. Available: https://creativecommons.org/publicdomain/zero/1.0/deed.en. [Accessed September 2020].

[18] "HDF view," HDF Group, [Online]. Available: https://www.hdfgroup.org/downloads/hdfview/. [Accessed July 2020].

[19] The HDF Group, "The HDF5 Library & File Format," [Online]. Available: https://www.hdfgroup.org/solutions/hdf5/. [Accessed June 2020].