Discriminate Human Interaction from Task Contact

handed in
MASTER'S THESIS

B. Sc. Felix Franzel

born on the 14.07.1995
living in:
Mozartstraße 41A
85579 Neubiberg
Tel.: +49 176 42000526

Human-centered Assistive Robotics
Technical University of Munich

Univ.-Prof. Dr.-Ing. Dongheui Lee

Supervisor:	Thomas Eiband
Start:	18.11.2020
Intermediate Report:	18.01.2021
Delivery:	18.03.2021

TECHNISCHE UNIVERSITÄT MÜNCHEN
Human-centered Assistive Robotics
UNIV.-PROF. DR.-ING. DONGHEUI LEE

February 11, 2021

MASTER'S THESIS

Discriminate Human Interaction from Task Contact

Problem description:

It has been shown in literature that robots are able to discriminate if a physical contact is a desired interaction or unwanted collision [3]. This work considers only contacts with the environment but no physical contacts with a human. Instead, we want to focus on situations, where the robot can come into contact either with the environment (task contact) or with the human in order to interact with the robot (human interaction).
In detail, we want to consider different situations which benefit from a classification between human interaction and task contact. During the learning phase, task contact and unintended human interference might occur alongside kinesthetic teaching. In the execution phase, the robot gets into task contact, interacts with a human, or interferes with the human in an unintended way, which can be a collision. A combination of task contact and human interaction has already been analyzed in a preceding work [4]. Different detection schemes (model-based [2] and model free) to discriminate between task contact, intended human interaction or unintended interference shall be compared.

Tasks:

- Literature survey on contact classification in human and task context [1]
- Development of features for a context aware classification algorithm
- Implementation and comparison of classification schemes (classification based, knowledge based)
- Experimental evaluation of your results on a real robot

Bibliography:

[1] Catherina Burghart, Sadi Yigit, Oliver Kerpa, Dirk Osswald, and Heinz Woern. Concept for human robot co-operation integrating artificial haptic perception. In *Intelligent Autonomous Systems*, volume 7, pages 38–45, 2002.
[2] T. Eiband, M. Saveriano, and D. Lee. Intuitive programming of conditional tasks by demonstration of multiple solutions. *IEEE Robotics and Automation Letters*, pages 1–1, 2019.
[3] Saskia Golz, Christian Osendorfer, and Sami Haddadin. Using tactile sensation for learning contact knowledge: Discriminate collision from physical interaction. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3788–3794. IEEE, 2015.
[4] Matthias Schoeffel. Exploiting internal and external force-sensing in human robot interaction. Technical report, Technical University Munich, 2019.

Supervisor:	M. Sc. Thomas Eiband
Start:	18.11.2020
Intermediate Report:	18.01.2021
Delivery:	18.03.2021

Felix Franzel

(D. Lee)
Univ.-Professor

Abstract

This work introduces a Contact Event Pipeline to distinguish Task-Contact from Human-Robot interaction during task execution. The increasing need for close proximity physical Human-Robot Interaction (pHRI) in the private, health and industrial sector demands for new safety solutions. One possible approach is the robust classification of contacts between human and robot.

A solution is designed, that enables simple task teaching and accurate contact monitoring during task execution. Besides an external force and torque sensor, only proprioceptive data is used for the contact evaluation. Two approaches are designed to distinguish contact events from normal execution. A contact type classifier, based on a Support Vector Machine, is trained with the identified events. The system is set up to quickly identify contact incidents and enable appropriate robot reactions.

Simulations are conducted with data recorded from intended and unintended contacts as well as examples of task-contacts like manipulating and interacting with surrounding objects. The Contact Event Detector and the Contact Type Classifier are evaluated separately for both approaches. Special focus lies in the robustness of the classification as well as the fast detection of contact events . The performance is evaluated on different experiments, including a real world task.

Contents

Chapter 1

Introduction

Collaborative robots or Cobots are a minor but steadily increasing sector in robotics. The International Federation of Robotics (IFR) predicts a Compound Annual Growth Rate between 50% and 60% [ifr18b]. Collaborative robotics is traditionally part of the service robotic sector and increasingly used for research purposes. Currently, it is progressing to be a more considerable option for industrial automation. With the typical industrial robot working in evacuated workspaces, there is a steadily increasing demand for more and more tasks to be performed with the joined effort of human operator and robot. The driving factors for the increasing number of Cobots are: Falling component prices, improvement of technical gear as well as sensors and research advances in artificial intelligence [ifr18b].

The collaboration of robot and operator in industrial surroundings can be of different forms. The IFR specifies four levels of collaborative applications in the field [ifr18a]. When ordered with increasing amounts of collaboration, the first is mere Coexistence. No boundaries are separating the workspaces of human and robot, but they do not overlap. Both partners perform their work separately form each other. When work is done sequentially in an overlapping workspace, it is defined as Sequential Collaboration. Robot and human can work on the same workpiece but not simultaneously. When the operator is working on the workpiece, the robot is at rest. To increase the efficiency, both partners can work simultaneously, which is the called Cooperation. In an even more advanced step, the robot responds to the movements of the worker. This is defined by [ifr18a] as Responsive Collaboration. The most frequently used types of collaboration in todays industrial surroundings are Coexistence and Sequential collaboration. But, to increase the range of possible applications and to further increase the economic value of cobots, more Cooperation or even Responsive Collaboration must be introduced.

Safety of the worker, such as summarized by the ISO 10218-1:2011 standard, is thereby the most pressing objective. Different perspectives of safe human robot interaction are analyzed in numerous prior works. Where some analyze the possible

injuries or inflicted pain [PHSA11, HHK+12, YHH+97], others propose safety strategies [IIN03], actuation mechanisms [ZRKS04], interaction control schemes [BT04], collision detection [EH02], reaction strategies to collisions [HADH08] or even complete contact avoidance during interaction [HDOEW19]. In general, when humans want to work with robots in close proximity, especially when direct interaction contacts are possible, complete safety cannot be guaranteed without sacrificing productivity. In industrial surroundings most machines still are safety hazards when not operated with caution. Robots are no exception to this fact. When robots are being used, a trade-off must be found between usability and total safety. For example, by reducing the executed forces and velocities to a minimum, almost complete safety can be guaranteed. But this makes the robot unprofitable for the factory use. Unintended contacts can therefore not be completely avoided if normal speeds and forces are kept. A monitoring system must therefore classify such unwanted interaction and avoid severe injuries. These monitoring systems can be camera or motion sensor-based [HDOEW19, EH02], tactile-based [GOH15, HADH08, KDA18] or a combination of both.

The advantages of using robots for production are straightforward: Robots can produce highly accurate motions for long periods of time. They can operate at very fast rates while being able to exert high forces. If servicing is neglected, they can be operated continuously without needing a break. Their advancing sensors enable them to analyze objects in an extremely detailed way. In contrast, humans still have a far more advanced visual perception and scene understanding. Humans are considerably quicker to understand the context of a task and develop strategies to reach their goal. Besides that, they have excellent fine motor skills. They are quick to understand logical connections and find solutions to complex problems. In combining the best abilities from both partners, collaborative robotics can automate tasks, enhance productivity and decrease long term health issues of tasks traditionally being done by human workforce. Many of those tasks also impose physical and mental stress on the worker. Among the most occurring symptoms are chronic back injuries [CPC+16]. Besides having significant health benefits, cobots increase productivity. Only the most coordinative demanding production steps are left to the worker, while more simple and monotonous steps are taken over by the robot. The most prominent industrial sector for the application of cobots is the assembly line. Among several possibilities, Kildal et al. [KTFM18] also list machine tending, inspection, welding, deburring and palletizing. With the increasing public focus on environmental conservation, also dis-assembly is a newly emerging sector for cobots. Another certain future goal is to incorporate cobots to assist healthcare workers [ifr18b]. Generally, cobots will be introduced in fields where qualified work personnel is increasingly hard to find.

Besides safety, flexibility is the second most important requirement for collaborative robotics, according to industry professionals in the statistics of [KTFM18].

Frequent changes in the production lines are increasingly important since products are rapidly updated and highly specialized [ifr18b]. The goal is to reduce the long period of time needed for system integration, installation, programming and parameter tuning. The setup of new production processes must be designed in a more intuitive way, for even less skilled workers to be able to perform these steps of re-tasking [ifr18b]. The lack of knowledge is one of the main barriers for the usage of cobots [KTFM18]. Therefore, more intuitive ways of programming and usage must be developed. This will open up new markets for end-users who are less familiar with robots. Even the usage in the private sector is possible. Highly versatile and flexible robots will become essential multi-purpose service assistants in home applications [HDA17]. It will be necessary that tasks can easily be trained, without extensive knowledge in robot programming. When tasks are "run" by the user, intuitive interaction with the robot must be enabled. A monitoring system, running in the background, should be able to analyse all contacts. The robot must be able to differentiate between contacts that are occurring because of the task, which is often called task-contact, and those that are initiated by the user.

For Sequential Collaboration like in [CPC+16], human interaction is often confined to the workpiece. Interaction with the robot arm itself, on the other hand, occurs more natural to the operator. Physical Human Robot Interaction (pHRI) mimicking the physical interaction between humans is more intuitive. The teaching process for a robot task is therefore often done by kinesthetic teaching. In more conservative robot collaboration applications, the worker must be briefed in which way interaction is allowed. It is thus more desirable to enable all intuitive types of pHRI. Examples are:

- Guidance of the robot when a task is not executed correctly
- Deformations of the trajectory to avoid obstacles in the path
- Hindering of the robot's movement when the operator is in the way

Because of these examples, a direct classification of intended user interaction is needed for the robot to be able to react accordingly. All the mentioned examples of intended contacts are initiated by the human. But likewise, contacts can happen in unintended ways if the operator is not minding the robot's movement or if the robot behaves in an unintended way. These collisions, of many possible types, must be quickly identified to guarantee the workers further safety. A collaborative human-robot system must thus be able to identify all possible human interactions and classify their type.

In summary, the growing market shows that there is a need for highly versatile and flexible industrial cobots. There is a growing number of tasks still performed by human personal, which can be done more efficiently with close proximity collaboration of human and robot. Appropriate safety measures must be taken for this

type of close interaction. This essentially means that a monitoring system must always be aware of the state of contact the robot is in. Whether this is done by camera, motion capture, tactile monitoring or artificial skin. The more sensors are used for this purpose, the more complex and expensive the system gets. But these systems are also not without faults, problems like occlusion can occur. According to [MMP+10, ifr18b], high technological flexibility is a key economic factor. High amounts of sensor tuning for very specific tasks like in [CPC+16] are inefficiently time-consuming. Furthermore, discrimination between contacts due to the task and contacts due to engagement with the human body must be performed. This is essential for the applicability of the monitoring system. Unnecessary reactions to contacts resulting from task engagement would make the system unreliable in industrial surroundings. Lastly, such a monitoring system should be able to operate in any given situation, without the need of extensive reprogramming for different tasks.

1.1 Problem Statement

It is therefore the goal of this work to provide a system that can distinguish task-contact from human interaction during Human-Robot Cooperation. It is desirable to develop a framework that enables simple recording of robot tasks by way of Learning form Demonstration (LfD) via Kinesthetic teaching, making it easily adaptable to various automation problems without extensive knowledge of robot programming. The system should not be confined to specific types of tasks, a small set of movements or very specific types of contacts. It is the main goal of this work, to provide a system that monitors the task execution to enable safe and productive pHRI. Therefore, contacts must be analyzed towards their detectability and in what way they can be distinguished from normal execution through the given sensor data. A method must be found and tested that can accurately and quickly classify a number of different contact types from said data. By way of identification and classification the system should recognize when and how the robot is interacting with the environment, the workpiece or the operator. The only sensors needed are the robot's proprioceptive sensors and an externally mounted force torque sensor. Finally, with a quick detection and accurate classification of the contact type, the goal is to enable the system to provide a reaction strategy that is appropriate to the identified and classified contact in a further step. The main focus must therefore lie on the robustness and accuracy of the classification and on the time delay between the detection of and the reaction to a contact.

1.2 Related Work

When robots are used likewise by trained and untrained personal, safe and intuitive interaction are the key features. The system must be able to detect and evaluate the state of contact between the robot and the human. Furthermore, the robot must react to certain contacts appropriately. We now give a resume on the prior work. We analyze how contacts are defined and how contacts are classified. Work concerning reactions to dangerous contacts are discussed. We also mention different solutions to the problem.

In their study, Burghart et al. [BYK+02] provide both a classification for human robot co-operation as well as for robot contact classification. The developed framework first distinguishes between a robot behavior that either does or does not include a further object to be manipulated. Without an object, the robot can be led or restricted by the human or the robot restricts the human. In our research, the robot can be interacted with or restricted by the user, with and without an object. The meaning of the term "object" is not restricted to movable objects only. Objects can also be fixed in the environment e.g. screws. Situations may also include the robot acting on parts of the environment with certain tools. Tools fastened to the robot are not counted as objects. On the other hand, regarding co-operations including an object, the object can either be handed over or simply be manipulated together. We will focus on objects that are manipulated by the robot together with the human in a symmetric way or totally independent of the human. A symmetric interaction thereby describes a situation where the human and the robot perform the same actions on the same object simultaneously. We interact with the robot while it is manipulating fixed and movable objects. The robot also interacts with the environment and movable objects by itself.

Furthermore [BYK+02] classify human-robot and environmental contacts into three main classes depicted in Fig. 1.1: collision, control contact and task-contact. A collision is, by their definition, a non-voluntary contact between a human and a robot or an unintended contact of the robot with the surrounding environment. Thereby, it is not closer defined if the robot interacts in a dynamic way with the human or by clamping the user in an unintended way. The purpose of a control contact is to give commands to or to correct the robot. In our case, control contacts will always be initiated by the operator. The defined sub-class of positioning could be for example a correction of the robots end effector in a placement task with the operators hand. Command inputs on the other hand, might be carried out as taps on the robots arm, to signal the robot to take up the task again after a stop. Finally, task-contacts are contacts that are, as the name suggests, related to the actual task of the robot. Where control contacts in our case are solely initiated by the operator, task-contacts are mostly initiated by the robot. In our context, task-contact will mainly be handling objects in a predefined way or intended contact between the tool

and the environment to facilitate a task. With the help of certain decision criteria Burghart et al. [BYK+02] form a classification model which helps to classify a certain robot contact type. This model is a decision tree which decides by the operation mode and the expectation of the robot, as well as the status and the intention of the user. In the following, we will stick to this designed classification and use the mentioned terminology.

Collision	Control Contact	Task-Contact
▪ Non-voluntary (human-robot) contact ▪ Unintended (robot-environment) contact	▪ Give commands to the robot ▪ Correction of the robot	▪ Related to robot task ▪ Initiated by the robot

Figure 1.1: Contact types as defined by [BYK+02].

It can be seen by the vast amount of research in this field, that estimating the contact state of the robot is a critical issue not only for safety reasons but also for close cooperation and task completion. For that reason, in [HDOEW19] contact between the human and the robot is totally avoided in order to obviate collisions at any time. In essence, they create an artificial repulsive force field around the human operator. It changes the planned trajectory to push the robot away from the human body at any time. Additionally, to guarantee a fast reaction, the human motion is predicted by Probabilistic Movement Primitives. By using black-box optimization, the weights of the used Dynamic motion primitives are updated with respect to the human interaction. A complete avoidance of physical contact however is not desirable for our framework, since control contacts could not be enabled. We also do not use any motion tracking sensors.

Instead, a framework is needed that can classify contacts while executing a task and react accordingly. Golz et al. [GOH15] propose a framework that distinguishes between intentional and unintentional physical human-robot contacts. The classification is done with non-linear support vector machines that are trained with features from the robots joint torque sensory data. By considering insights from a physical contact model, their simulation results show the possibility to even discriminate between collisions with different body parts. Despite mentioning four different contact classes, distinguished by the robots contact with the environment and the intention of the contact, they do not include task-contact at all e.g. contact with a manipulatable object, or other types of task-contact mentioned above, into their framework. Their classification is confined to intended (control) and unintended (collision) contacts.

On the other hand, Haddadin et al. [HDA17] present a survey on different algorithms for collision detection. All algorithms rely solely on proprioceptive information. While most mainly monitor the external torques applied to the joints via different computational methods, collisions are detected with the help of a function comparing a moment observer with an experimentally evaluated threshold. In an isolation step, the affected link and associated contact point is estimated. A prior assumption is made, that only a single link can be affected. In the final identification step, the collision joint torques and/ or the Cartesian wrenches at the contact point are calculated. Those three steps are part of a seven-step collision event pipeline, consisting of a pre-collision state, detection of the contact, isolation of the contact point, identification, classification, reaction and post-collision. We chose to not include any pre-contact measures. Thus, we do not impose restrictions on the workflow by reducing the speed or altering the robot trajectory in order to avoid contact. Such precautionary measures are computationally expensive for online application and need additional external sensors. To us, only three of the mentioned are relevant. Those are: Detection, classification and reaction.

Lastly, in [KDA18] contact situations are analyzed with the help of the Spectral Norm Derivative of force sensor data measured on the end effector or in the robot joints. Advantageously, this approach does not rely on any specific model with a prior training phase or data recordings of example interactions. Rather, a sliding window of force measurements is taken into consideration. The fast Fourier transform is applied to the windows time series and the spectral norm derivative is computed from the spectrum. A threshold is experimentally evaluated by which the contacts are classified. High frequency changes within a specific margin thereby indicate undesired collisions, whereas lower values could hint at voluntary collaboration. Even though this method is considerably faster than compared methods in detecting unexpected collisions, it works best for collisions with high frequency changes, velocity and energy transfer. Another disadvantage is the distinction threshold between intended and unintended human-robot interactions. The threshold must be picked and can vary between execution examples. Furthermore, no task-contact can be evaluated by this method.

Additionally, some research is performed on expanding the variety of sensor data. By using multiple instead of a single sensory modality, Park et al. [PEBK16] state that a wider variety of deviation from the normal execution can be detected. During task execution, haptic, visual, auditory and kinematic sensor data is evaluated. Hidden Markov models are trained with likewise non-anomalous sensory readings. Additionally, a varying log-likelihood threshold which indicates faulty behavior during the execution is utilized. This threshold varies over time and is execution progress dependent. Evaluation on complex tasks shows a better performance compared to unimodal anomaly detection.

Task-contact on the other hand is only studied as a separate subject, mostly in relation to assembly tasks. A. Rodriguez et al. [RBM+10] study the force signature of assembly tasks. They identify successful and failed industrial production processes with Support Vector Machines. Their supervised date-driven approach does not require prior knowledge of the specific task for classification of the result. Rather, by compressing the force signatures with Principal Component Analysis only a small amount of training examples is needed and therefore a new task can easily be trained. Also, as stated by them, a single load cell which is properly aligned provides enough discriminative information to classify the attempts. With this approach, the force signals could be monitored online with the system set up to detect erroneous force time-series at an early stage.

As can be seen from the preceding work, no classification has been designed that includes control contact along with collisions and task-contact. The aim of our work will therefore be to come up with a solution for a task monitor that can perform online detection of the three contact classes. After prior kinesthetic teaching, the robot should be able to perform a collaborative task in close proximity to the human without posing a safety hazard. While for example performing collaborative assembly tasks, the robot must be able to detect collision at an early stage, while not falsely classifying a task-contact with the environment as such. Additionally, during the task execution, the possibility to physically and intuitively control the robot must be given to the human operator. This control contact must thereby be clearly distinguished from the two other contact forms, in order to prevent injury.

Chapter 2

Technical Approach

In this chapter we will in detail describe a system to distinguish human-interaction from task-contact. For this purpose we will introduce a Contact Event Pipeline. It will be explained what kind of data is used and how the system is set up to achieve this goal. Two approaches will be compared on how to detect and distinguish between contacts. Lastly, it will be shown how the proposed system is implemented.

2.1 Design of Solution

The primary target of this work is to discriminate human interaction from task-contact at any time instance the robot is executing a predefined task. Learning from Demonstration via Kinesthetic teaching is used to teach tasks to the robot. During this teaching phase, human and robot are in direct contact for the majority of the time. The robot is guided through the motion that is to be executed later. Additionally, the robot is in the state of gravity compensation, which means that it cannot exert any force or momentum on the teacher by itself. It solely follows the forces applied to it and freely follows along the given path. Simultaneously, all sensors record data. An external force and torque sensor, applied in between the last link of the robot and a two-finger gripper, records forces and torques in three Cartesian directions. At the same time, the Cartesian position of the end effector with the associated rotation matrix, the angular position of each robot joint and the joint motor torques are recorded. Mathematically, the gravitational force influences of the gripper mass on the external force and torque sensor are compensated. The saved force/torque sensor data then represents only the real exerted forces and torques. This is done with the raw sensor data in what is generally known as a force-torque compensation and based on the work of [KKW07]. The position of the center of mass of the gripper as well as the mass itself and the bias of the force-torque sensor are estimated by driving the robot into predefined positions. Likewise, an estimate of the external joint torques $\hat{\tau}_{\mathbf{ext}}$ in each robot joint can be calculated by subtracting the internal model dynamics from the joint motor torques $\tau_{\mathbf{joint}}$ in (2.1).

If the manipulator (2.1) is controlled by the impedance controller (2.2), an estimate of the external torques can be computed as:

$$\boldsymbol{\tau}_{ext} = \boldsymbol{K_j}(\boldsymbol{q_d} - \boldsymbol{q}) + \boldsymbol{D_j}(\dot{\boldsymbol{q}}_{\boldsymbol{d}} - \dot{\boldsymbol{q}}) + \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_{\boldsymbol{d}} - \ddot{\boldsymbol{q}}),$$

We assume that the following assumptions hold:

- $\tilde{\boldsymbol{M}} \approx \boldsymbol{M}$, $\tilde{\boldsymbol{c}} \approx \boldsymbol{c}$, $\tilde{\boldsymbol{g}} \approx \boldsymbol{g}$
- $\hat{\boldsymbol{h}} \approx \boldsymbol{h}$ is a close estimate of the real internal friction and damping.

Dynamic model of the manipulator:

$$\boldsymbol{\tau}_{joint} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\tau}_{ext}, \tag{2.1}$$

where $\boldsymbol{q}$ is a vector of joint angles, $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix, $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ are the Coriolis and centrifugal torque, $\boldsymbol{g}(\boldsymbol{q})$ is the gravitational torque and further torques $\boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ that are resulting from damping and friction.

Impedance control:

$$\boldsymbol{\tau}_{com} = \boldsymbol{K_j}(\boldsymbol{q_d} - \boldsymbol{q}) + \boldsymbol{D_j}(\dot{\boldsymbol{q}}_{\boldsymbol{d}} - \dot{\boldsymbol{q}}) + \tilde{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \tilde{\boldsymbol{c}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \tilde{\boldsymbol{g}}(\boldsymbol{q}) + \hat{\boldsymbol{h}}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{2.2}$$

where $\boldsymbol{q_d}$ denotes the desired joint angular positions, $\boldsymbol{K_j}$ and $\boldsymbol{D_j}$ are the stiffness and damping parameters and $\tilde{\boldsymbol{M}}$, $\tilde{\boldsymbol{c}}$, $\tilde{\boldsymbol{g}}$ are the internal model of the corresponding mechanical terms, where $\hat{\boldsymbol{h}}$ is an estimate of the damping and friction.

Generally, there are two main phases in which the robot can have task-contact or human interaction. The first being the described physical teaching phase and the second being the phase of executing the taught procedure. During the teaching phase no harm can come to the teacher from the robot. Considering the three main classes of contact, defined in the introduction, only control contacts and task-contacts (no collisions) could occur. The discrimination of task-contact and human interaction during this teaching phase would more likely be interesting in the face of analyzing the effects of environmental contact versus human contact, which we do not focus on at this point in time. We rather prioritize the phase in which the robot reproduces the taught movements. As discussed, during this execution of the learned task, the most harm can come to the human. Especially when working in close proximity or in direct contact with the robot. If the operator is incautious or does not know the task, the robot reproduces a motion that could lead to a collision with the human or a clamping situation. The entire process must therefore be monitored continuously and any harmful situation must be eradicated as quickly and efficiently as possible.

Regarding this premise [HDA17] introduce a Collision Event Pipeline. Its focus lies on the specifics of the sensor data. It extracts and evaluates specific details of collision events, which results in very specific reactions to the collisions. In contrast, we introduce a Contact Event Pipeline (CEP) with some similarities to the one mentioned before. The main differences are that more than just one contact class is defined and that we propose a data driven approach that does not require a hand tuned threshold based on a momentum observer. Furthermore, our more extensive detection and classification system still has comparable small delay. Haddadin et al. [HDA17] mention a reaction time of 200 ms for one of their approaches. Specifically, we introduce two more contact types that can occur in industrial settings and during human robot interaction. Besides the involuntarily collision, we desire to detect planned or constructive human-robot interaction. Additionally, we plan on detection and classifying contact resulting from the robot task. In accordance with [HDA17] a detected collision should eventually lead to a stop of the robot motion. But the additional contact types must be detected while continuing the task. The full CEP is depicted in Fig. 2.1. From here on forward, a contact or contact event will be defined as everything that deviates from a robot movement in free air. This includes contacts of the robot with its surrounding objects as well as the human operator.

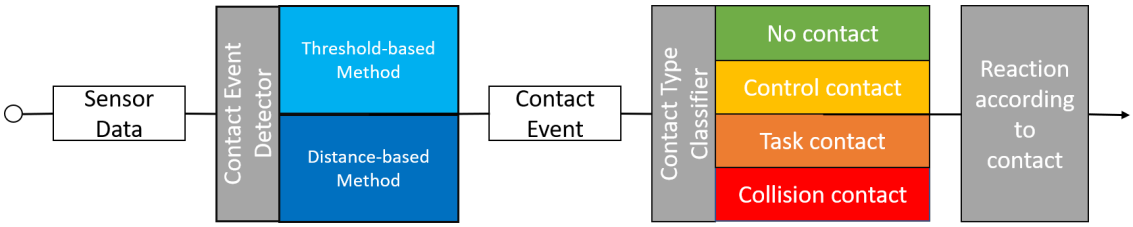


Figure 2.1: Contact Event Pipeline for the detection, classification and reaction to contact events in robotic scenarios.

Considering the appropriate reaction to each of the three types of classifiable contacts (task-contact, control contact and collision), one must find a good trade-off between the robustness of the classifier and the time that will pass until the robot reacts. It is trivial that, especially for collisions, the time duration must be as short as possible i.e. taking into account as few consecutive sensor readings as possible. While on the other hand evaluating as many consecutive sensor readings as possible to analyze the contact and maximize the reliability of the classification without sacrificing the proceeding of the task due to a false classification. Both, the trade-off between quickness and accuracy and the linkage between classification and reaction will be discussed later on.

Other than [HDA17], we do not incorporate a Pre-contact Phase into our Contact Event Pipeline out of two reasons: First, all methods to avoid or minimize

the impact of contacts need additional external sensors that monitor the robot's environment. We only use the robots proprioceptive sensors with the addition of an external force torque sensor. Secondly, besides being computationally expensive [HDA17] many prior strategies change the trajectory or its parameters or try to avoid any contact at all. Additionally, [HDA17] state that even with the extensive robot motion planning algorithms it is not possible to guarantee collision-free behavior at the moment, due to the quickness of human motion.

The first step in the Contact Event pipeline is the detection of a contact event. The output of this step is a simple boolean information, if at the current time instance a contact event occurred or is still ongoing. This information is decided by evaluating the current measured sensor data. At this point, two approaches are compared on this matter. The Threshold-Based Method and the Distance-Based Method, as can be seen in Fig. 2.1. The Contact Event Detector (CED) extracts all the sensor readings $\boldsymbol{x}_t = [\boldsymbol{p}_t, \boldsymbol{o}_t, \boldsymbol{\tau}_t, \boldsymbol{w}_t, \boldsymbol{q}_t] \in \mathbb{R}^{26}$ that are characterized as being part of the contact event. Where $\boldsymbol{p}_t = [x, y, z] \in \mathbb{R}^3$ is the Cartesian end effector position, the orientation either in euler angles $\boldsymbol{o}_t = [\alpha, \beta, \gamma] \in \mathbb{R}^3$ or unit quaternions $\boldsymbol{o}_t = [q_w, q_x, q_y, q_z] \in \mathbb{R}^4$ depending on the approach, $\boldsymbol{\tau}_t = [\tau_1, ..., \tau_7] \in \mathbb{R}^7$ the external torques in each joint, $\boldsymbol{w}_t = [f_x, f_y, f_z, t_x, t_y, t_z] \in \mathbb{R}^6$ the wrench from the external sensor and the joint angles $\boldsymbol{q}_t = [q_1, ..., q_7] \in \mathbb{R}^7$at each time stamp t.

In a second step, the Contact Type Classifier (CTC) uses only the data samples characterized as to be part of the contact event. They are classified as one of the contact classes. Instance for instance the entire contact is classified. This is done with a Support Vector Machine (SVM). The predictors are features computed from a sliding window of sensor readings $\boldsymbol{X}_{t,...,t+N} \in \mathbb{R}^{N\text{x}26}$, where N is the length of the sliding window.

One of the main reasons for choosing SVMs for classification is that they are considered to be one of the most robust and accurate among the supervised machine learning algorithms [BG12, GOH15]. Since for our recorded data, sample points related to the same contact class are not in a compact cloud, the use of e.g. K-Nearest Neighbor would not be wise. On the contrary, Support Vector Machines maximize the hyperplane separating the class representing data points, which can be of any given form and direction. Gaussian kernels are used to project the data into a higher dimensional feature space incorporating a non-linear SVM classifier to further enhance the performance. Despite taking a long time for the optimization process, SVMs are among the fastest in classification. This is essential for online classification. Additionally, they work well with large numbers of features compared to the number of training instances while being less prone to overfitting than other methods [BG12]. Since it is not always the case that each feature is a valuable contribution at every time, SVMs offer another valuable attribute.

They are considered to be "excellent" in tolerating irrelevant attributes and "very good" in tolerating redundant attributes [BG12].

We use features derived from the sensor data to classify contacts. This is more likely to succeed compared to using the sensor data directly, since different samples of contact events within a contact class can differ immensely. Features, on the other hand, are more likely to capture specific aspects of the data and generalize more within a contact class. Additionally, sign and direction of the contact become irrelevant. Golz et al. [GOH15] derive the maximal resulting contact force in a contact event (collision or control contact) between a robot and a human as

$$f_{\text{ext}}^{\max} = \sqrt{\frac{M_R M_H}{M_R + M_H}} \sqrt{K} \dot{x}_R^0,$$

where M_R and M_H are the reflected inertia of robot and human respectively. K is the effective contact stiffness which is mainly given by the human contact area in the case of a rigid robot [HASH09] and $\dot{x}$ is the relative impact velocity. Respectively, the maximal contact forces in the case of introducing task-contacts besides results in:

$$f_{\text{ext}}^{\max} = \sqrt{\frac{M_R M_X}{M_R + M_X}} \sqrt{K} \dot{x}_R^0$$

where M_X represents either the human or the environmental contact stiffness. It is made clear by [GOH15] that the signal cannot effectively be represented by linear features alone. This is due to several reasons, among which are the non-linearity of the sensing process or the eigenfrequency that factors into the resulting contact force. We therefore choose for most of our features to be nonlinear. Their mathematical expression can be seen in Tab. 2.1. The first twelve features are adapted from [GOH15]. The Hjorth Complexity along other features is a widely used feature in touch modality identification via tactile sensing [KLC15, BAA15]. Other features are not adopted due to reasons of small informative value or long processing time. It has been found out during testing that some features used by [GOH15] do not produce values that are correlated to the type of contact. Other features are computationally very expensive, which makes them unusable for real time application. Additionally, if the computation of one feature prolongs the entire process, it can not be applied.

Additionally, we incorporate the features 13-17. When it comes to distinguish between task-contact and collision, an important aspect can be the general movement properties in that instant. Therefore, features of velocity and acceleration are included. Kouris et al. [KDA18] draw the line between collision and control contact based on the rate of change of a predefined frequency band. The Spectral Norm Derivative (SND) is the time derivative of the 1-norm of a predefined frequency band defined by an upper and lower bound. The computed SND-value is compared

to a threshold to detect collisions. The SND is used as a feature, with the difference that the considered frequency band in the positive spectrum is defined only by an upper bound $\omega_{\max}$. This is done to reduce the influence of sensor noise. Frequencies of task contact and control contact lie in the lower bands, therefore the lower frequency bound is set to zero. To give the model an idea of which joint is mainly involved in the contact event and in what way, the last features represent the physical work executed in the joints and the external sensor. This indicates in what way the environment or the human is acting on the robot or vice versa. The work feature indicates if energy is transmitted from the robot to the environment or from the environment to the robot.

The effectiveness of the selected features is evaluated with two selected algorithms: The ReliefF [RŠK03] and the Minimum Redundancy Maximum Relevance (MRMR) algorithm [DP05]. With the MRMR we analyze if any of the chosen features is irrelevant. The algorithm maximizes the relevance

$$V_S = \frac{1}{|S|} \sum_{x \in S} I(x, y)$$

of the feature set S with the number of features $|S|$ by considering the mutual information

$$I(X, Y) = \sum_{i,j} P(X = x_i, Y = y_j) \log \frac{P(X = x_i, Y = y_j)}{P(X = x_i) P(Y = y_j)}$$

between the feature X and the class label Y, while minimizing the redundancy

$$\Lambda_S = \frac{1}{|S|^2} \sum_{x \in S} I(x, z)$$

of the set by computing the pairwise mutual information between the predictors X and Z. A mutual information value of zero implies that X and Z are independent. Additionally, the robustness of the features is analyzed by the ReliefF algorithm, which uses a nearest neighbor type approach. Predictor weights are computed by rewarding if the k-nearest neighbors of a certain observation are also within the same class. The weight is reduced if these neighbors belong to a different class.

After all features are calculated from the training samples, they are normalised along each of the feature dimension to the range of [0 1]. This avoids that features with high values are favoured during the training. All features are computed from $\boldsymbol{X}_{t,\ldots,t+N}$. Velocity and acceleration are derived once per time instance t. When calculating all features over a total of seven joints, three external forces and tree external torques, a total of 184 features are computed per time instance. Only one Support Vector Machine model is trained for the entire Robot. As opposed to [GOH15] where one model represents each one senor dimension or joint respectively,

we propose to use only one model over all sensor dimensions out of many reasons. The process proposed by [GOH15], trains many individual models, subsequently lets each model predict a contact class with associated posterior probability upon receiving a contact instance and finally calculates an overall probability for every contact as the mean of all probabilities for each contact class. This process does not represent the robot involved in the contact instance thoroughly. Additionally, this takes a lot of time for computation. Most importantly, not every sensor or joint contributes the same amount of discriminative information. Golz et al. [GOH15] propose to adapt factors by which the individual model results should be weighted for specific robot configurations or tasks. But this would sacrifice the generality of the entire process and simultaneously increase the time for manual programming for individual tasks. Some kind of reference table would have to be set up, adapting the weights for each individual task configuration. One model on the other hand combines the wide range of sensors and features respectively taken from the entire physical robot and models the contact instance in its whole complexity.

The SVM model is trained and k-fold cross validated to find the best fit for the hyperparameters. The model is also able to return posterior probabilities during a classification. With the help of the trained model and the described process in this section, a new task can be trained from demonstration and subsequentially executed while contact instances are classified instantly and simultaneously. A detail description of the specific implementation of the entire process can be found in Sec 2.3. With the contact class being decided on, a reaction best fit to the contact can be executed as the last step of the Contact Event Pipeline.

Feature List		
f_1	Mean	$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$
f_2	Standard Deviation	$\sigma = \frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2$
f_3	Maximum	$\max(x)$
f_4	y-Range	$\max(x) - \min(x)$
f_5	Median	$\widetilde{x} = \text{median}(x)$
f_6	Hjorth Complexity	$C(x) = \frac{\frac{\sigma_{dd}}{\sigma_d}}{\frac{\sigma_d}{\sigma_x}}$
f_7	Shannon Entropy	$H(x) = -\sum_{i=0}^{m} p(x_i)^2 \log(p(x_i))^2$
f_8	Energy	$E_s = \int_{-\inf}^{\inf} \|x_t\|^2 \, \mathrm{d}t$
f_9	Skewness	$s = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})}{\sigma^3}$
f_{10}	Autocorrelation	$a = \frac{\sum_{i=1}^{N-\kappa}(x_i - \bar{x})(x_{i+\kappa} - \bar{x})}{\sum_{i=1}^{N-\tau}(x_i - \bar{x})^2}$
f_{11}	Phase	$\text{Phase}(\text{FFT}\ (x)) = \arctan\frac{\text{Im FFT}\ (x)}{\text{Re FFT}\ (x)}$
f_{12}	Amplitude	$\text{Ampl}(\text{FFT}\ (x)) = \frac{\sqrt{[\ \text{Re FFT}\ (x)]^2 [\ \text{Im FFT}\ (x)]^2}}{N}$
f_{13}	Velocity	$v = \left\| \frac{\partial(x)}{\partial t} \right\|_2$
f_{14}	Acceleration	$a = \left\| \frac{\partial(v)}{\partial t} \right\|_2$
f_{15}	Spectral Norm Derivative	$I_{\text{SND}} = \frac{\partial \left\| \boldsymbol{f}_{\omega_{\min};\omega_{\max}} \right\|_1}{\partial t}$
f_{16}	Work in Joints	$W_J = \sum_{t=t_1}^{t_2} \frac{\partial A_t}{\partial t} T_t^T$
f_{17}	Work in Sensor	$W_S = \sum_{t=t_1}^{t_2} \frac{\partial [P_t, O_t]}{\partial t} W_t^T$

Table 2.1: List of all features used for contact classification with mathematical formulation.

2.1.1 Possible Reaction Strategies

Since fully executing the trained task is the main goal of the whole operation, the appropriate reaction to a classified task-contact would most naturally be the further execution of the task. Detecting a control contact means that the human wants to alternate the robots trajectory, orientation, improve the robots motion or any other reason mentioned in the introduction. The robot should then cooperate with the human in the most natural way. This means that the impedance of the impedance control should be drastically decreased, as to reduce the forces exerted on the operator. A switch to admittance control is also possible. After the control contact is lost, the robot should proceed with the task by possibly updating the trajectory, forces, torques, etc. But this has not been applied at this point in time. For collisions, De Luca et al. [DAHH06] propose the best reaction to be a "bounce back" in the direction that mostly minimizes the contact forces while coming to a complete hold once the contact forces have reached a less dangerous level. This kind of reaction is especially most favorable in the case where collision is of the type that it is clamping the human. The direction of the most relief can be drawn from the external joint torques which indicate the amplitude and direction of the resulting contact. A threshold value, regarding the estimated joint torques, can be defined, at which the "bounce back" is stopped. To continue the task after a collision, when safety for the user is regained, applying pressure on the robot exterior for a short moment is a generally used signal. A more simple measure is to bring the robot to an instant stop and discard the rest of the motion.

2.2 Detection Methods

In this section, the specifics of the two proposed event detection methods outlined in the previous section will be described. In the following, it will be discussed how the data is recorded, how samples are labeled and how the features are extracted from the training samples. Special focus lies on the different implementation of the two approaches for contact event detection. We will elaborate on how the SVM model is trained and how it performs on testing samples. First, we start by explaining the process for the Threshold-based Method and go on to discussing the differences to the Distance-Based Method.

For this work, the sample data is recorded from running the robot through individual tasks for every contact class. The example trajectories are trained by Kinesthetic Teaching. Each example is recorded by guiding the robot through free air and without interacting with the environment. It is important to mention that samples for each contact class are recorded separately, i.e. one recording exclusively for each contact type. It is desirable for the taught examples to be unique in their way of movement. A set of examples representing one contact class should span a wide range of motions. To record samples for control contact, the respective examples are run, while the operator interacts with the robot during the execution. Again, these interactions should comprise a most wide spectrum of pushes and pulls in each possible direction, at each joint and on the tool. The more extensive the range of contacts are, the more robust the prediction will be. The same postulate holds for the following two contact types. Collision contacts are recorded similarly by placing a dummy in the trajectory of the robot arm. To record task-contacts, the robot trajectory is either altered in Cartesian directions, e.g. for the robot to get in contact with objects in the environment, or objects are placed in predefined positions, e.g. to be picked up or moved by the robot. For instance, one exemplary trajectory is altered in negative z-direction to get in contact with a table, in order to simulate a sliding movement of the end-effector or a tool on a surface.

2.2.1 Threshold-Based Method

The Threshold-Based Method (TBM) for finding a contact event is based on monitoring the estimated joint torques and the external sensor readings for non-normal amplitudes. The normal region is defined by sensor values drawn from the example trajectories being executed in free air. Therefore, the data only contains forces and torques exerted by the robot and tool itself together with sensor noise. Since the example movements consist of motions in various orientations, the data contains a valid set of data for every joint and sensor dimension. This set of "normal executions" is taken to calculate reference values. Mean and standard deviation are calculated from the stacked example recordings. A time instance is flagged as con-

tact event if one dimension of the associated sensor reading exceeds a given threshold $[\boldsymbol{\tau}_t, \boldsymbol{w}_t] > \boldsymbol{\mu} + k * \boldsymbol{\sigma}$ where $\boldsymbol{\tau}_t$ and $\boldsymbol{w}_t$ are the estimated joint torques and external sensor data as previously indicated and $\boldsymbol{\mu}, \boldsymbol{\sigma} \in \mathbb{R}^{13}$ are the mean and standard deviation vectors drawn from the example trajectories. The factor $k \in \mathbb{R}$ is evaluated from training samples in a preliminary analysis. A trade-off must be found between avoiding false positives of no contact situations, which can occur due to high sensor noise, and triggering correct flags as soon as possible when needed. The value $k = 8$ is found to produce accurate results. A full list of all used parameters can be found in Tab. 3.1 in the Chap. 3.

The remaining false positives are hand labeled in a preprocessing step. Data samples for each of the contact classes are labeled during the recording. The Contact Event Detector finds each data sample $\boldsymbol{X}_t$ exceeding the threshold. Start and end of the contact events are defined by hand by considering the output of the CED. Likewise, the regions of false positive contact events are indicated. The associated contact label is derived from the label given to the sample recording. The classification is based on a constant number of samples following the detected instance $\boldsymbol{X}_{t,\dots,t+N}$. Here, the length of the sliding window N must be found to ensure an accurate classification but must be chosen short enough to guarantee the quickest possible reaction. With a sampling time of 2 ms and a set containing 85 samples the resulting computational reaction time of 0.17 s is considerably short. For each time instance that is detected by the CED such a set of samples is passed to a function calculating the features. The term chosen for the set of samples is monitoring window. For the autocorrelation the shift κ is 10 and the upper bound $\omega_{\max}$ for the SND is 110. The features are calculated for all 13 sensors, one feature at the time. The resulting features are stacked into a combined features vector $\boldsymbol{F}_t = [\boldsymbol{f}_{1,t}, ..., \boldsymbol{f}_{17,t}]$ with each $\boldsymbol{f}_i \in \mathbb{R}^{13}$ except for the velocity f_{13} and acceleration f_{14} which are $f_{13,14} \in \mathbb{R}$. Each set of features is labeled according to the label of the respective sample.

2.2.2 Distance-Based Method

In contrast to the Threshold-Based Method, this approach uses task-knowledge encapsulated in the reference sample to detect deviations from the desired procedure resulting from human interaction. All internal and external sensor measurements are compared, instance for instance, to an existing example of the same task. Deviations in the robot's state or movement as well as the exerted forces are considered as contact by the CED. To make this approach fully autonomous, an algorithm is used to automatically find the start and end point of contact events to avoid hand labeling during preprocessing. From the findings of this algorithm, a threshold is derived for online contact event detection.

Contact events for this Distance-Based Method (DBM) are found by comparing a reference task recording with the executed task. This is done by computing a distance metric over the reference sample $\boldsymbol{y}_t = [\boldsymbol{p}_t, \boldsymbol{o}_t, \boldsymbol{\tau}_t, \boldsymbol{w}_t, \boldsymbol{q}_t]$ and the executed recording $\boldsymbol{x}_t = [\boldsymbol{p}_t, \boldsymbol{o}_t, \boldsymbol{\tau}_t, \boldsymbol{w}_t, \boldsymbol{q}_t]$. The distance metric measures the dissimilarity between $\boldsymbol{x}$ and $\boldsymbol{y}$:

$$d(\boldsymbol{x}_t, \boldsymbol{y}_t) = \sum_{k=0}^{n} (x_{t_k} - y_{t_k})^{1.5}, \quad n = dim(\boldsymbol{x})$$

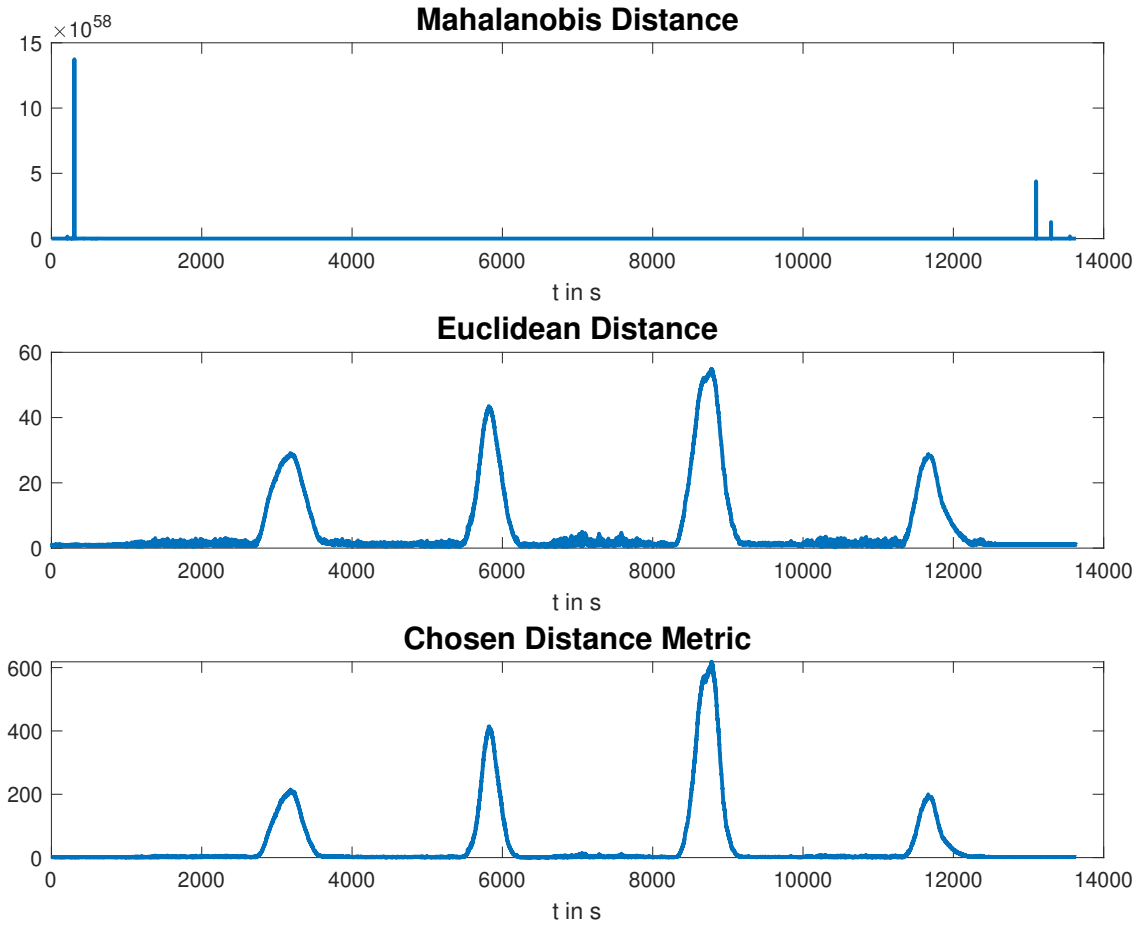


Figure 2.2: Comparison of different possible vector distance metrics for contact event detection. Mahalanobis distance with a sliding window of N samples as reference (top), Euclidean distance (middle) and our chosen distance metric (bottom).

For this approach, the orientation in unit quaternion formulation is used since they are continuous. This distance metric is chosen among several possibilities because it is real time compatible and it delivers the most accurate representation of the dissimilarity of the samples. In a preliminary analysis several possible distance metrics have been compared. Among them are the Mahalanobis distance, several forms of

the Minkowski distance (L_p-norm) and mean square error. For real time compatibility, the sliding window of the compared samples for the Mahalanobis distance is bound by the length of the sliding window N for the feature computation. A comparison of some of the tested vector distance metrics is depicted in Fig. 2.2. It can clearly be seen that the Mahalanobis distance does not work well in real time with a small set of N samples to compare an incoming measurement to. Furthermore, it reacts poorly to changes in dimensions with high data variance and favors changes in dimensions with low variance. The Euclidean distance shows more noise in between contact events compared to our chosen distance metric. This can aggravate the process of contact event discrimination. The chosen distance metric both smoothes the areas between contact events and avoids high variance during the a contact event.

An accurate distance threshold at which a dissimilarity is counted as a contact must be found. Additionally, in this approach we want to avoid hand selecting contacts from the samples. To find contact events from the distance vector over the whole length of the sample T , a changepoint detection algorithm is used which finds rapid changes in a signal [Lav05, KFE12]. More accurately, it partitions the signal into sections by minimizing the total residual error between the signal and a predefined statistical property.

The residual error is given by:

$$J(K) = \sum_{r=0}^{K-1} \sum_{i=k_r}^{k_{r+1}-1} \Delta(d_i; \chi([x_{k_r} ... x_{k_{r+1}-1}])) + \beta K \ , \tag{2.3}$$

where k_0 and k_K are the first and the last sample of the signal, χ denotes the empirical estimate and Δ is the measured deviation. A penalty is added for an increasing number of changepoints K and by the proportionality factor β.

A common mathematical procedure is to maximize a log-likelihood instead of minimizing the error.

The joint log-likelihood function of N independent observations is given by:

$$\log \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x_i-\mu)^2/2\sigma^2} = -\frac{N}{2}(\log 2\pi + \log\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} (x_i - \mu)^2 \tag{2.4}$$

where μ and σ are the mean and the standard deviation of a normal distribution of the observations.

The empirical estimate can be one of a statistical property e.g. mean or standard deviation. A comparison of all possible statistical properties has proven the root-

mean-square level to deliver the most accurate results in finding the start and end of a contact event even with more than one contact event in one sample recording. We chose the property to be the root-mean-square level, which uses a constant mean set to zero and a piecewise constant variance. With this setting the changes in the variance of the distance are analyzed. By maximizing the log-likelihood with respect to the variance and a fixed mean (2.4) reduces to [Lav05]:

$$\underset{\sigma}{\operatorname{argmax}}\, l(0, \sigma^2) = (n - m + 1)\log\sigma^2$$

where:

$$\sigma^2 = \frac{1}{(n - m + 1)} \sum_{i=m}^{n} (x_i - \mu)^2$$

In essence, the second sum of (2.3) is then formulated by:

$$\sum_{i=k_r}^{k_{r+1}-1} \Delta(d_i; \chi([x_{k_r} ... x_{k_{r+1}-1}])) = (n - m + 1)\log(\frac{1}{n - m + 1} \sum_{i=m}^{n} x_r^2)$$

The number of changepoints can be chosen by hand. The penalty term is therefore statically increased towards K, minimizing the risk of too many changepoints. Figure 2.3 shows an example plot of the distance metric over one control contact trajectory recording with the changepoints indicated by vertical lines. It can be noticed that contact event is encapsulated by the boundaries.

The essential drawback of this algorithm is that it does not work in real time. Although it achieves accurate results for the preprocessing of the training samples, another method must be found for real time classification. Preferably, one using a related method. The range of possibilities includes taking a mean of the distance metric at the changepoints as a threshold. Similarly, a threshold comparable to the Threshold-Based Method can be drawn from data not considered to be part of the contact by the detector. Another alternative would be to monitor again a sliding window of samples, where the residual error of the first half is compared to the error of the second half. The method chosen is to take in to account the mean of all distances at the indicated changepoints and use them as a threshold.
Analogous to the first TBM approach, a set of N samples are used for the computation of the features. This window of constant size is slid over all samples lying between the changepoint boundaries.

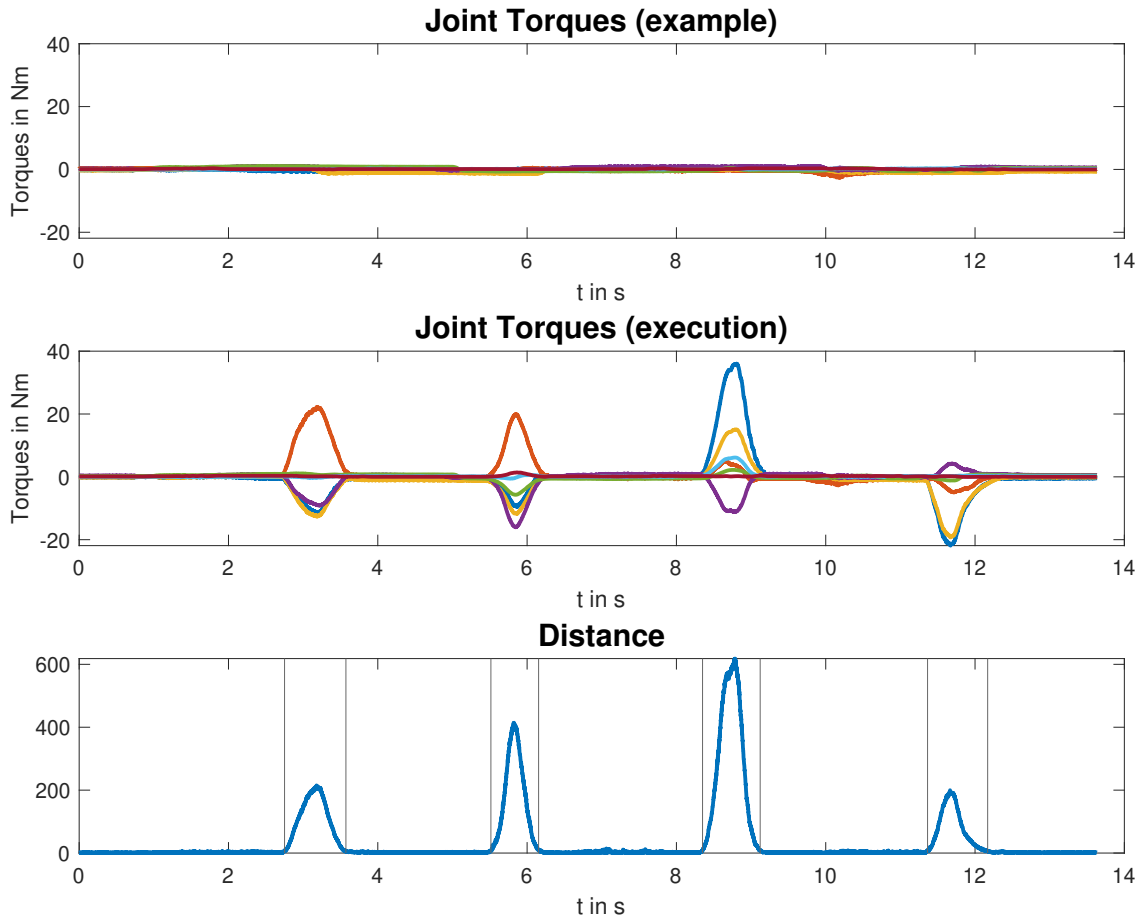


Figure 2.3: Distance metric based analysis of control contact recording, indicating the found change points with vertical lines. Two following changepoints enclose one contact.

The essential difference between the two methods is the strategy of treating task-contact. In the TBM, task-contacts are classified along the other possible contact forms if a contact event is triggered. For the DBM, task-contacts are part of a normal execution and therefore do not trigger the CED. We call this the task knowledge of the system. Task-contacts are part of the reference sample taught by kinesthetic teaching and once run without human interaction. Therefore, only deviations like control contact or collisions lead to a significant increase in the distance and subsequentially trigger the CED. This makes the kind of task-contact irrelevant to the classification. The SVM reduces to a classifier that decides between collision or control contact during contact events. The models of both approaches are capable of filtering false positives among the detected contact events. These false positives are labeled as "no contact" by the CTC. In summary, the label of "no contact" is given to instances that are labeled as not part of any contact event by the CED, as well as to false positives that are classified as such by the CTC.

2.3 Implementation

In this section, the specific implementation of the process outlined in the previous section will be described. In the following, it will be discussed how the data has been recorded, how samples are labeled and how the features are extracted from the training samples. Special focus lies on the implementation of the two approaches for contact event detection. We will elaborate on how the SVM model is trained from the training samples and how it is used in real time. First, we start by explaining the process of training the SVM and the differences between the Threshold-based and Distance-Based Method. Later we go on to explaining the real time implementation for experimental usage.

The implementation of the approach is generally divided into two distinct parts: the offline training of the model and the real time implementation for experimental purposes. The sampling, labeling, training of the model and simulation has been conducted on MATLAB while the real time script is implemented in C++/ROS.
A key difference between the two proposed methods, distance and threshold based, is the implementation of the labeling process. While for the distance-based method processes and labels the samples automatically, labeling for the threshold-based method must be done by hand. This process works as follows: A function using the determined contact threshold passes the sections believed to be a contact to the user, who then decides if the particular section is indeed a contact or no contact. The type of contact is then automatically determined from the description of the sample. Hence, while recording the samples only one type of contact has been performed. For the distance-based method, the additional information needed is the number of performed contacts. This integer and the distance between the sample and the reference data is then passed to a function finding the sections of contact by way of using the changepoint algorithm. Both these functions return a list of contact boundaries indicating the sections of contact.
Next, the sample data with the information about the contact sections and the type of contact are passed to a function which computes the features from the sample data. Following that, the returned features are normalized and passed to a function training the model. Pseudo code clarifying this process is shown in Alg. 1.

The real time implementation of the system uses and collaborates with the robot interface. Therefore, the compatibility with the robot interface cycle time is one of the most important aspects. Key processes like the detection of contact events and the computation of the features are computationally inexpensive and can therefore be computed in the main loop. Opposed to that, a prediction of a contact type label has an average duration of 9.0 to 9.5 milliseconds which makes it not compatible with the 2-millisecond robot cycle rate. To solve this problem, the prediction of the contact type label is outsourced to a separate thread, detached from the main thread. This makes a contact type label available to the main thread at any time

while an update is performed roughly every 10 milliseconds without delaying the main process. For reference, the script is run on a Intel i5 3.30 GHz CPU. Pseudo code of this real time algorithm is depicted in Alg. 2.

Algorithm 1 Training of SVM

```
1: procedure TRAIN MODEL(method)
2:     Read sample data
3:     Set parameters                          ▷ Parameters for feature calculation
4:     switch method do
5:         case "TBM"
6:             contactBoundaries ← findContactEvent(data, threshold);
7:         case "DBM"
8:             distace ← getDistance(data, reference);
9:             contactBoundaries ← findContactEvent(distance, numContacts);
10:    for all contactBoundaries do
11:        features ← computeFeatures(data, contactBoundaries, parameters);
12:    end for
13:    model ← trainModel(features)
14: end procedure
```

For the communication between the separate threads, two global variables are defined, to store the features for the prediction and the contact type label returned by the prediction. After initializing the robot and the associated interfaces, loading the data for feature normalization and the reference sample as well as loading the SVM model, a separate thread is started to hold the prediction process. In the loop reproducing the trajectory, each iteration, the current robot state, estimated internal torques, Cartesian acceleration and velocity as well as the external wrench data is derived from the robot and sensor interface. This data is then passed to a function determining if the robot is currently in a contact state. In case of the TBM this contact-event-detector function uses the current readings of the reproduced sample as reference. If the robot is in fact considered to be in the state of contact, a contact flag is returned to be true and the features are computed from the current data and stored in the global feature variable. Simultaneously, the prediction function picks up the newly computed features, when ready, and predicts the contact type label using the trained SVM model. The type of contact is then displayed to the user via a dialog window on screen. The dialog window also contains a progress bar indicating the progress of the current task. If the predicted contact type label indicates a collision, the robot is configured to stop and end the process.

The influence of the tool to the raw data of the external force/torque-sensor is subtracted by a function using the current state of the end-effector, the tools weight and point of mass and sensor bias. The tools parameters can be determined and

compensated using $force\_torque\_tools$ from [Alm18] according to [KKW07]. The parameter estimation assumes that the robot can be controlled through MoveIt.

Algorithm 2 Real time implementation

```
1: global contactLabel
2: global features
3:
4: function predictionTread(model)
5:     contactLabel ← predictContact(model, features)
6: end function
7:
8: procedure Main:(method)
9:     System Initialization
10:    Get norm data                          ▷ For feature normalization
11:    Get reference sample
12:    Load model
13:    new tread ← predictionTread(model);
14:    while Robot running & !trajectoryFinished do
15:        reproduceTrajectory(reference);
16:        currentData ← getRobotData();
17:        currentData ← getFTSensorData();
18:        switch method do
19:            case "TBM"
20:                contactFlag ← contactEventDetector(currentData);
21:            case "DBM"
22:                contactFlag ← contactEventDetector(currentData, reference);
23:        if contactFlag then
24:            features ← computeFeatures(currentData);
25:            setUserInformationWindow(contactLabel);
26:        end if
27:        if contactLabel = "collision" then
28:            Stop robot and return
29:        end if
30:    end while
31: end procedure
```

Chapter 3

Evaluation

In this chapter the theoretical approach outlined in the previous chapter is evaluated towards its functionality. In detail, we will elaborate on how training samples are generated, how parameters for the contact detection as well as the feature calculation are derived and how the models are trained. The viability of the Contact Event Pipeline will be tested both in simulation and experimentally. More specifically, the two key elements, the accuracy of contact detection as well as the prediction of the correct contact type by the model, are tested. Finally, both methods will be compared by the outcome of their experimental results. A KUKA LWR-IV+ robot with a wrist mounted jr3 force/torque-sensor and a Weiss Robotics WSG-Series gripper is used for both the simulation and the experiment.

Initially, motion data is generated from kinesthetic teaching. The robot is put into gravity compensation and physically guided through the desired trajectory by the user. In total, 14 different types of movement have been recorded. Each movement consist of a unique set of motions to encapsulate a wide range of motions from real robotics tasks e.g. pick and place, pushing button, moving etc. It has been made sure that at least every rotation of all robot axes as well as different linear movement are among the set. This is to make sure that forces and torques of every kind can be applied on any joint in all directions. Once the set of representative types of movements is recorded, they are once run on the robot in free air. This set of "free-air"-movements solves two purposes. First, from this set the thresholds for the Threshold Based Method can be derived. The fact that the unique types of movements cover motions of every axes and that the tool applies its own gravitational forces to every dimension of the internal and external sensors, makes the resulting thresholds for every dimension most reliable. Therefore, the system is not prone to errors of falsely triggered contacts if it moves in a different way. Secondly, the "free-air"-movements are later used as reference samples for the Distance Based Method. Since no internal forces as well as reliable external forces on the force/torque sensor can be recorded during the kinesthetic teaching phase, this prior recording of task knowledge is inevitable.

For control contacts, five types of trajectories are recorded. As discussed, they include linear and angular motions in all possible directions to involve all rotary joints. Based on them, control contact forces and torques are applied on every rotary axis as well as on the tool. This will enable the user to perform control contacts on the entire outer surface of the robot and especially on the tool. Example trajectories for task-contact include four representative examples from the wide range of all possible task contact. The first consists of applying repetitive force on a horizontal surface. This could represent assembly operations, e.g. pressing parts together or applying pressure on an object. Another possible example could be the instance of contact between a screwdriver and the screw in a screwing task. The second involves applying pressure on and sliding along a horizontal surface (Fig. 3.7(b)). The other examples include the movement of small objects in different directions (Fig. 3.7(a)). The objects are moved only on the table surface and not lifted. For the recording of collision samples, a human like dummy, shown in Fig. 3.2 has been used. Five types of movement are chosen to impact the dummy on the arm, shoulder and head. This represents the human body parts that would most likely be affected in a real collision scenario with a single arm robot. For each type of contact at least two, but mostly four runs have been recorded. Each run contains multiple instances of contact. Only one type of contact is used for each run and the recordings are labeled accordingly. All recorded types of motions for every contact class are summarized in Fig. 3.1. Additionally, the total number of contact events per contact type is listed. Please note, while there are less contact events for the class of task-contact, some of these contact events have a considerably longer duration. This produces more classifiable instances per contact event compared to the other contact types.

	Collisions	Control Contacts	Task-Contacts
Number of recordings	12	9	8
Total number of contact events	35	33	22
Types of sample motions	▪ Collisions with arm ▪ Collisions with shoulder ▪ Collisions with head ▪ Impacts with robot arm and tool	▪ Linear and angular motions ▪ Control contacts on robot arm and tool	▪ Applying repetitive force on horizontal surface ▪ Sliding on surface motion ▪ Moving object

Figure 3.1: Total samples and contact events recorded for model training and simulation.

The recorded samples are split in half. The first half serves as training samples for the models while the second serves as testing samples during the simulation. In this way it is made sure, that for all of the example motions there is at least one recording that is unknown to the pipeline. For each type of movement there exists at least one training and one test sample but with unique contacts, since no two contacts are the same. Furthermore, all of the recordings consist of multiple contact events. Therefore, the training set consists of ample contacts to train the model on. The testing set is purposely kept as separate recordings and not partitioned in to the individual contact events. In this way, the simulation closely represents the execution of a entire task. Thus, the behavior of the pipeline during an experiment can be predicted.

Figure 3.2: Dummy for collision sample recording.

To generate the best possible result, first a parameter estimation is performed. The most fitting parameters are chosen to increase the performance of the Contact Type Classifier as well as the accuracy of the Contact Event Detector. All variable parameters are listed in Tab. 3.1. The length of the sliding window N for feature calculation, the delay κ for the autocorrelation and the upper frequency bound ω_{max} are chosen via 10-fold cross validation. Figure 3.3 depicts the misclassification rate in relation to the change of the evaluated parameters. The factor k for the TBM is tuned by hand by analyzing the sample data. It is the key factor of accurately distinguishing between contacts and their false positives like sensor noise. It is chosen to indicate a contact even early and the entire duration of the contact but

minimize the load to the Contact Type Classifier of filtering out the false positives i.e. "No Contact".

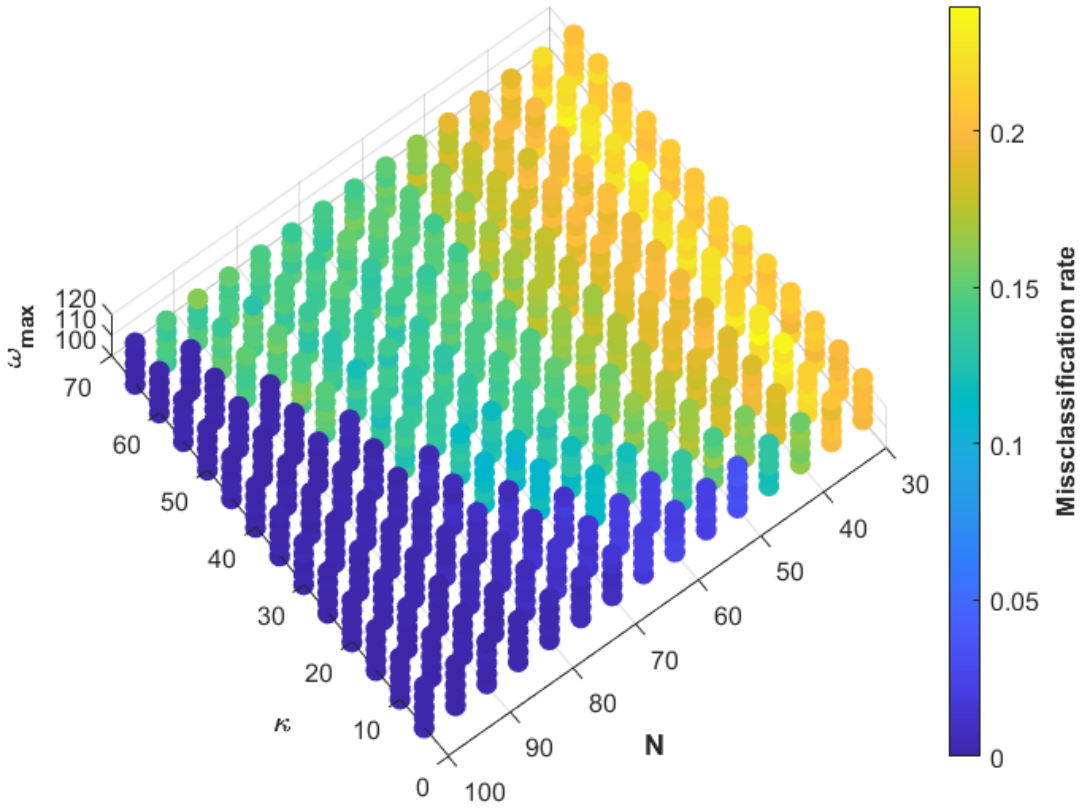


Figure 3.3: K-fold cross validation of parameters N, $\omega_{\max}$ and κ.

Parameter List	
κ	10
$\omega_{\max}$	110
k	8
N	85

Table 3.1: List of parameters used in the the computation of the features, and detection of contact events.

With the chosen parameters, the features are extracted from the training samples. The models for both methods are trained on the training set using k-fold cross

validation for hyperparameter estimation. The cross validated performance of the models is shown in the respective confusion matrices in Fig. 3.4.

		Predicted class			
		No contact	Control contact	Task-contact	Collision
True class	No contact	99.92%	0.08%		
	Control contact		99.43%	0.57%	
	Task-contact			99.98%	0.02%
	Collision				100%

(a) Confusion matrix TBM model

		Predicted class		
		No contact	Control contact	Collision
True class	No contact	99.91%		0.09%
	Control contact		99.65%	0.35%
	Collision	0.02%		99.98%

(b) Confusion matrix DBM model

Figure 3.4: Confusion matrices for SVM models of TBM and DBM for classification of No contact, Control contacts, Task-contacts and Collisions.

The computed predictor data is used to evaluate the chosen features according to the discussed methods: The ReliefF and the MRMR algorithm. The resulting feature importance weights and the associated feature rankings for the MRMR and ReliefF algorithm are depicted in Fig. 3.5 and Fig. 3.6 respectively. For the MRMR algorithm, a small drop in the weights indicates that the difference in the predictor importance are not significant. According to the ReliefF algorithm, only the acceleration features show a low level of robustness in the observations. By removing any of the less robust features, the model accuracy does not increase. The same accounts for a feature reduction with a principal component analysis.

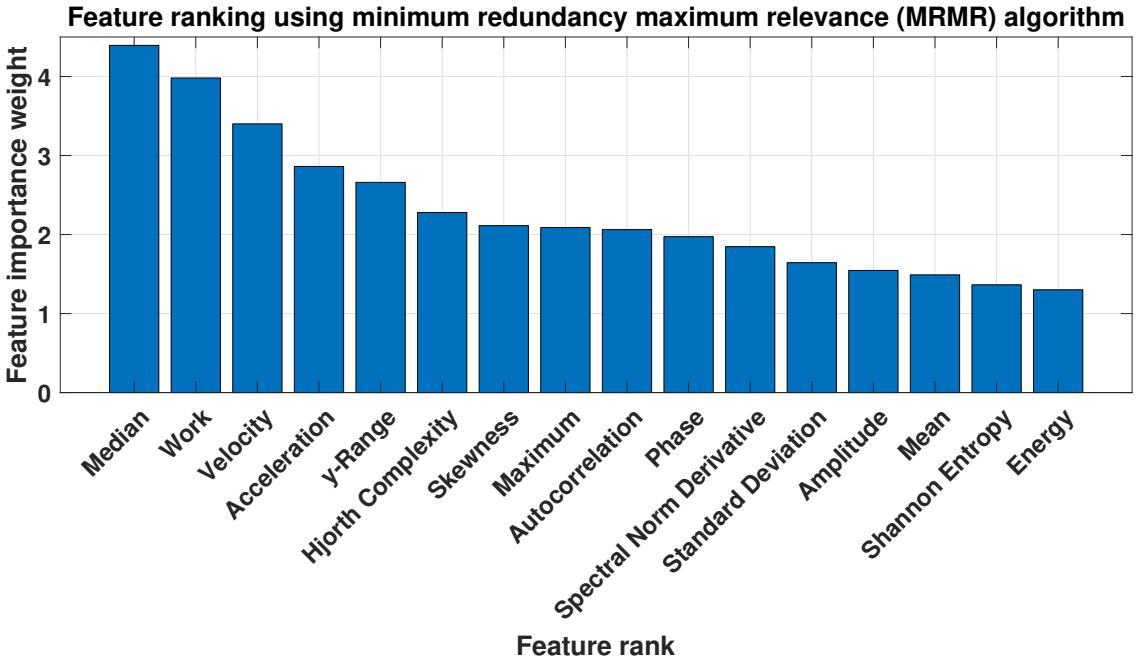


Figure 3.5: Results from the MRMR algorithm to determine relevance and redundancy of the chosen features.

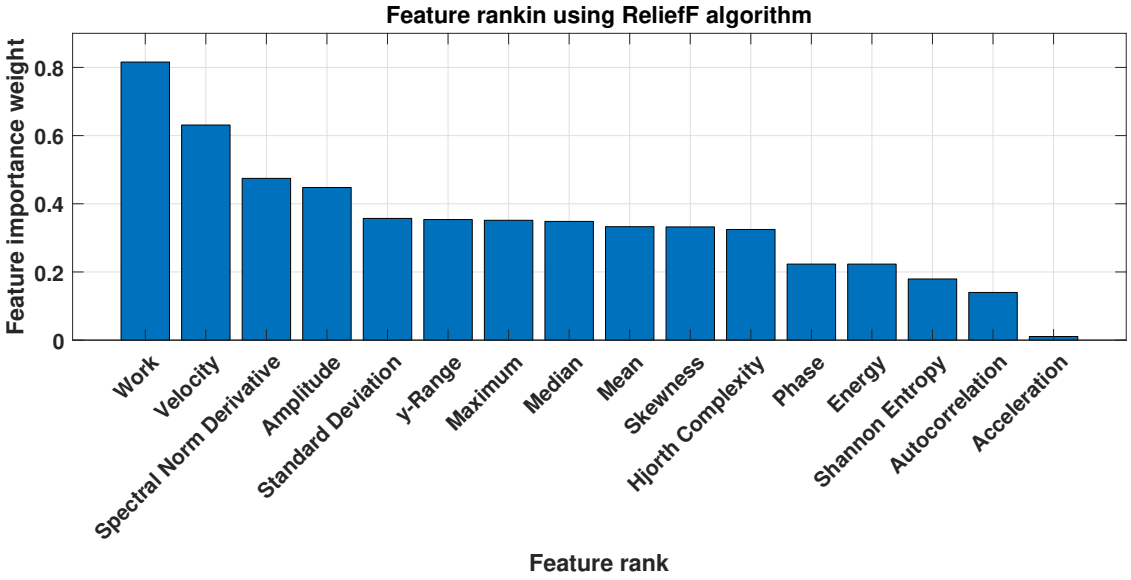


Figure 3.6: Results from the ReliefF algorithm to determine the robustness of the chosen features.

3.1 Simulation Results

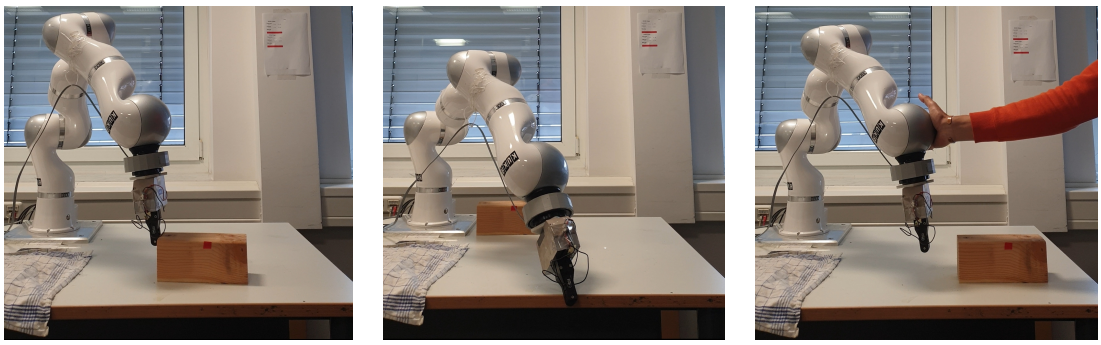
The simulation is conducted to prove that in principle both approaches, the threshold-based and the distance-based method, work on data samples from unknown contacts. The main goals of these tests are to show that first, both the Contact Event Detectors can accurately distinguish contacts from normal execution. This can be proven by analyzing the underlying plots of the joint torque and external sensor force/torque data. The second focus point is on the accuracy of the prediction from the Contact Type Classifier. The main aspects here are robustness of the classification i.e. if the entire duration of the same contact is classified as such. Also, how accurate are the initial instances of the contact classified, especially for collisions. Another aspect is the filtering performance of the CTC to find false positives among the contacts.

The simulation is performed with the data declared as testing samples. To simulate a real time implementation of the process, data samples of the recordings $\mathbf{x}_t$ are passed to the pipeline one instance at the time. This replicates how, in a real implementation, the data would be received from the robot and sensor interface. The instances are monitored as described in Sec. 2.2 and contact events are triggered by the CED. For the TBM the CED compares the incoming sensor data to the thresholds and triggers if they should exceed it. The CED of the DBM compares the sensor data together with the robot state to the reference data and likewise triggers if the maximum distance is exceeded. Blocks of data of the size defined by the contact window and indicated as part of the contact event are then passed to the CTC. The indication of contact events by the CED and the resulting classification by the CTC can be seen in the following figures. The following figures depict a comparison of the three or two contact types for the two approaches respectively.

3.1.1 Results of Threshold-Based Method

The first three figures show the result of the simulation for the Threshold-Based Method. The top two plots in Fig. 3.8 depict estimated joint torques $\mathbf{T}$ and wrench data $\mathbf{W}$ of a sample involving three collisions. Only the force data is depicted in the second plot for better visibility. The collisions can clearly be seen from the spikes of the sensor data. Only some of the trajectories show an increase. This is due to the fact that the forces resulting from the collision are in the direction of only some of the sensors dimensions. The high rate of change in the contact forces indicate a quick change of momentum, which is typical for a collision. The sharp peaks in the external sensor data indicate that the tool was only passively involved in the contact. Consequentially, the impact must have occurred above the external sensor. Additionally, rows four and five depict where a contact event occurred as decided by the CED and the resulting contact classification. The last row shows a plot of the contact class probabilities for all three possible classes. It can be noticed that the

initial part of the section declared as contact is always classified correctly (in this case collision). In the course of the contact the prediction switches to a task-contact. It can also be noticed that some of the instances declared as contact are predicted to be of the type "no contact". This means that the CTC filtered out some of the false positives from the CED. Similarly Fig. 3.9 involves five control contacts. The control contacts occur both above and below the external sensor. This is indicated by the fact that only two of the five contacts can visually be observed in the external sensor data. The rate of change in the force/torque data clearly distinguishes the previous collision from the control contacts. Most of the indicated contacts are classified correctly. Only minor irregularities in the classification occur at the end of the contacts. One coherent task-contact is depicted in Fig. 3.10. The related data is derived from the task of moving an object along the surface of the table. This is depicted in Fig. 3.7(a). Again, it can be seen that the tool has been involved in the action and that the trajectories have a unique structure. The whole duration of the task-contact is thereby classified correctly. Some of the rows three and four show a minor delay between the indicated contact event and the beginning of the contact classification (different from "no contact"). This can mean that the CTC did not classify the initial instances as one of the contact types.



(a) Task-contact object (b) Task-contact sliding (c) Task-contact object with interaction

Figure 3.7: Task-contacts as part of the training samples. Depicted are: Moving of an object (first), sliding on the edge of a table with the tool (second) and a control contact during the movement of the object (last).

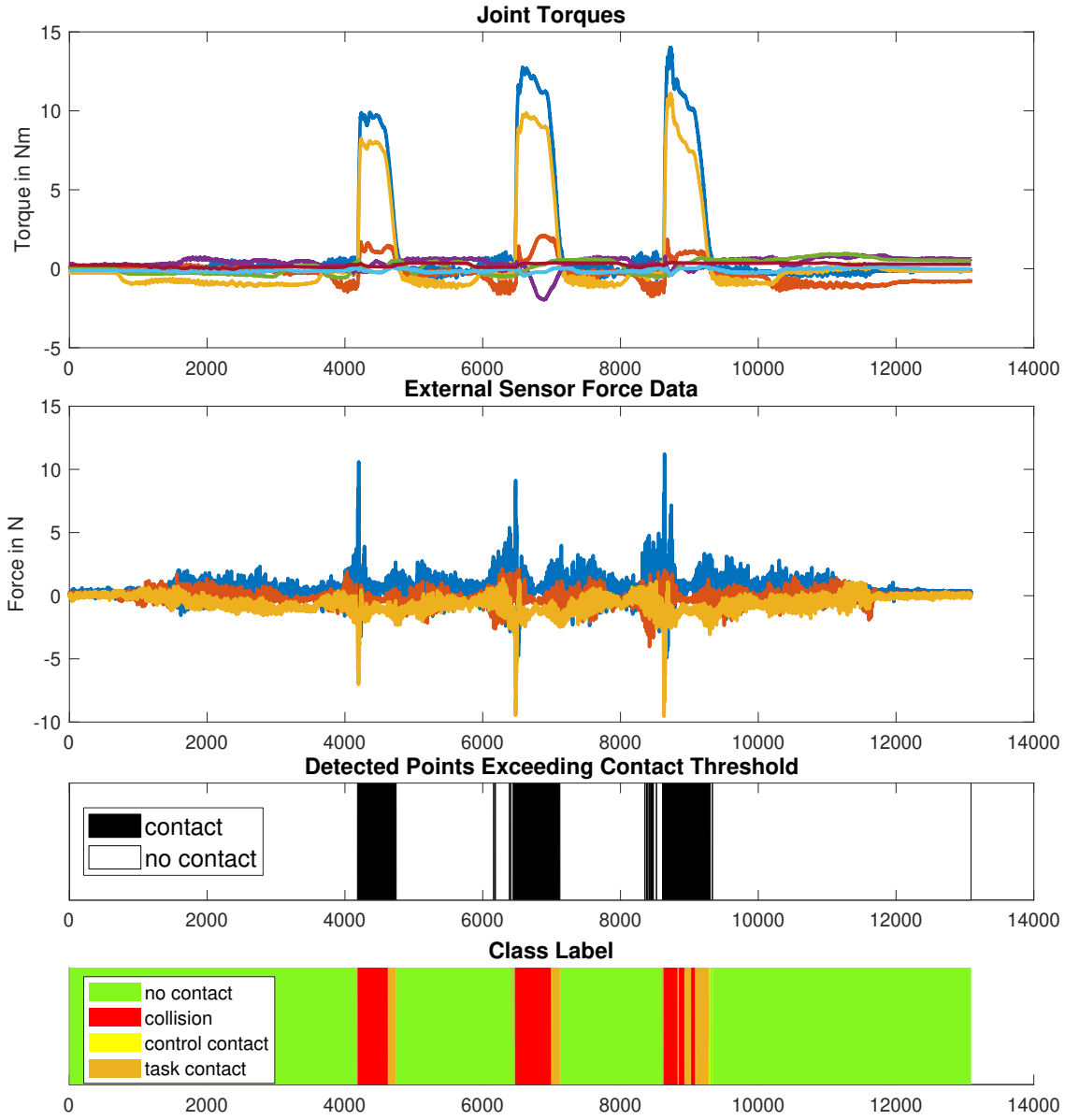


Figure 3.8: Simulation result of the CEP for TBM approach with three collisions occurring. The first two plots show the torque and force profile from the internal and external sensors. The last rows indicate points instances detected to be of a contact event as well as the predicted contact type.

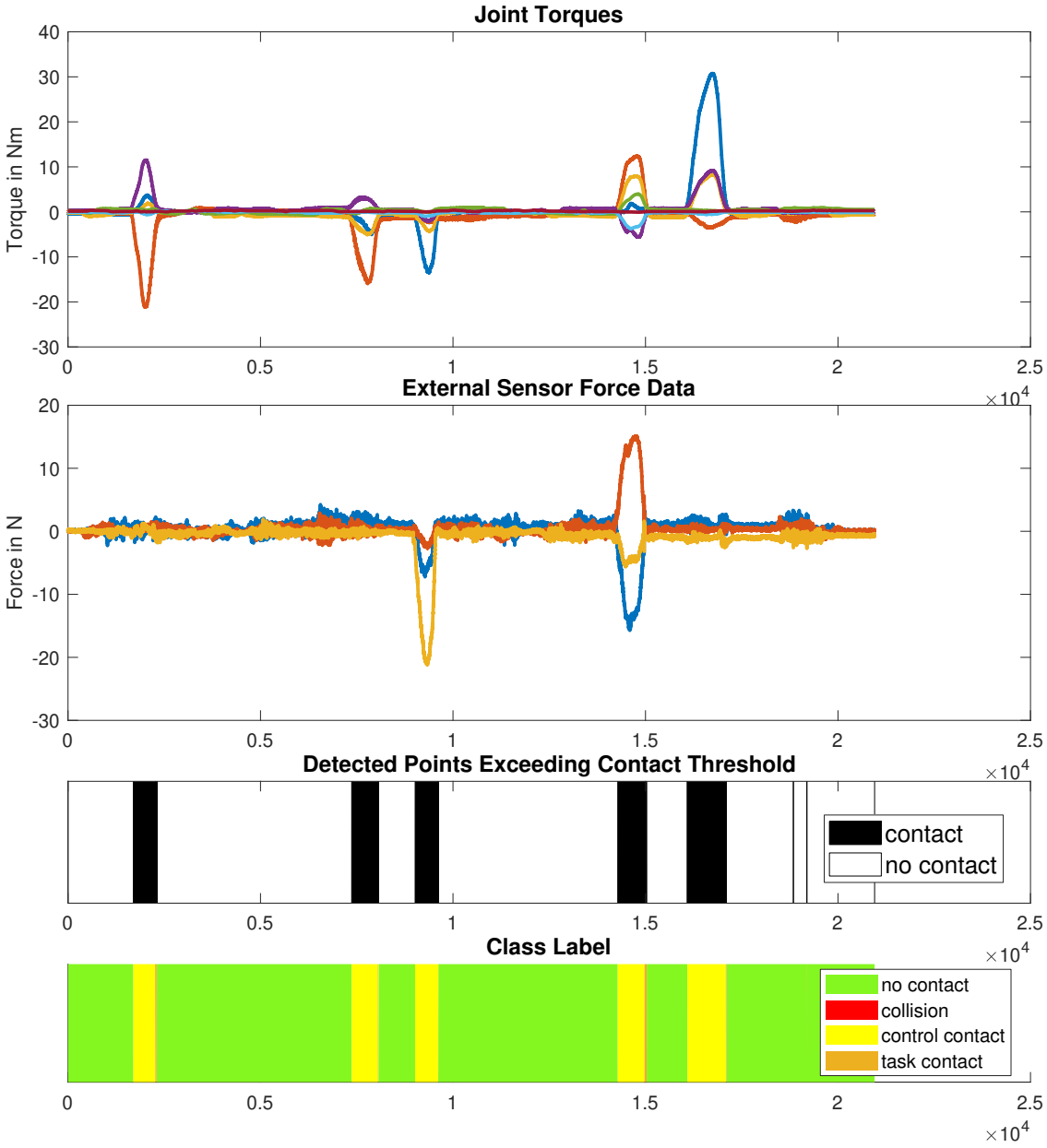


Figure 3.9: Simulation result of the CEP for TBM approach with five control contacts both on the robot surface as well as on the tool. The first two plots show the torque and force profile from the internal and external sensors. The last rows indicate points instances detected to be of a contact event as well as the predicted contact type.

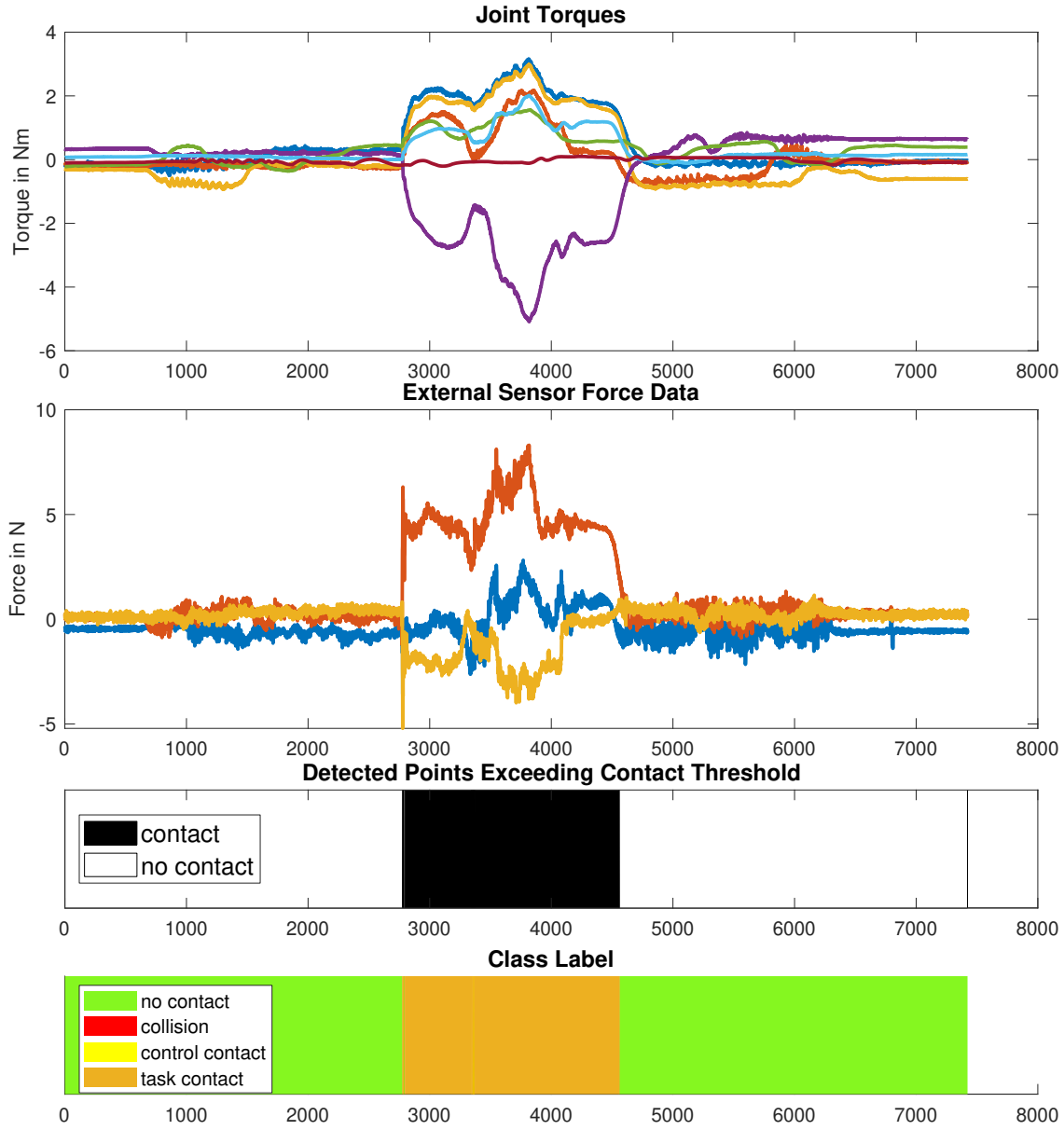


Figure 3.10: Simulation result of the CEP for TBM approach with one task-contact from moving an object. The first two plots show the torque and force profile from the internal and external sensors. The last rows indicate points instances detected to be of a contact event as well as the predicted contact type.

3.1.2 Results of Distance-Based Method

Figures 3.11 and 3.12 show the respective results of the DBM approach. Both figures include an additional plot. It indicates the total amount of the distance metric which is used to indicate contacts by the CED. Here, no task-contacts are shown, since as discussed, these are part of the task-knowledge or reference sample

and therefore cannot be classified. Again, the collisions are classified not entirely correctly but correct at the start of the contact. The control contacts are classified correctly throughout the whole contact.

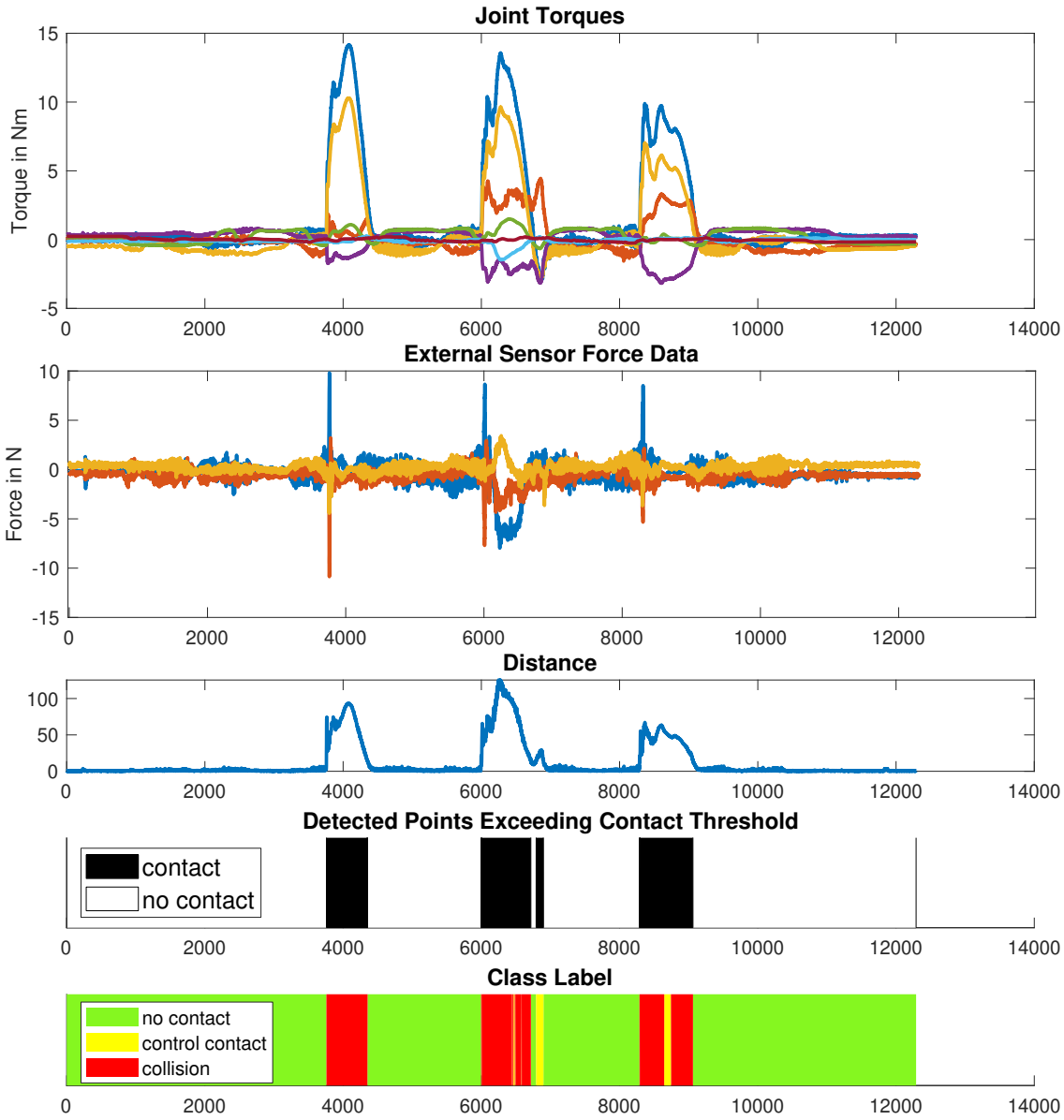


Figure 3.11: Simulation result of the CEP for DBM approach with tree collisions occurring. The first two plots show the torque and force profile from the internal and external sensors. The third plot shows the distance offset from the reference sample. The last rows indicate points instances detected to be of a contact event as well as the predicted contact type.

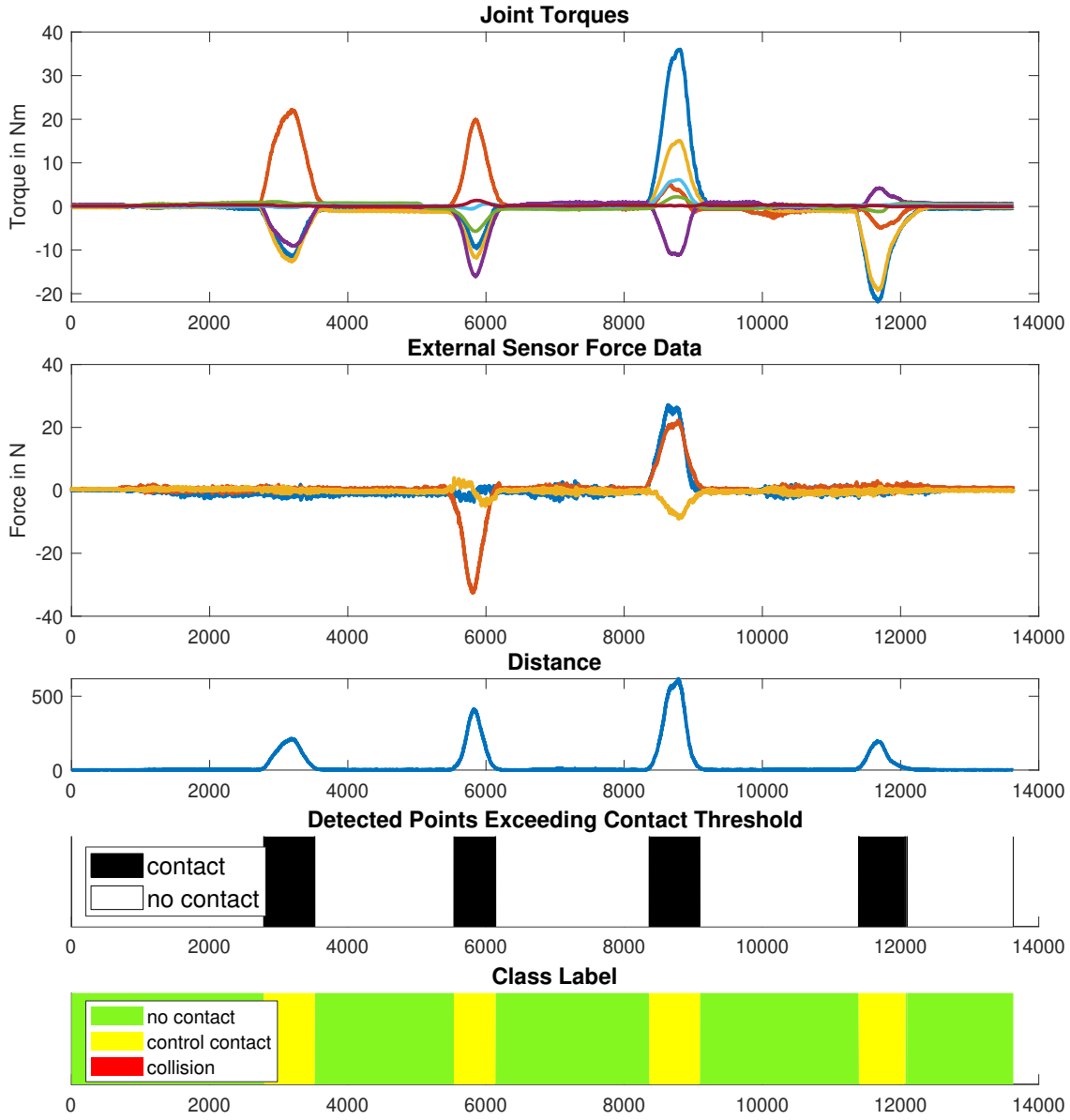


Figure 3.12: Simulation result of the CEP for DBM approach with four control contacts. The first two plots show the torque and force profile from the internal and external sensors. The third plot shows the distance offset from the reference sample. The last rows indicate points instances detected to be of a contact event as well as the predicted contact type.

3.1.3 Results of Combined Contacts

Figures 3.13 and 3.14 both show the simulation of combined contact types. For this test one of the example task-contact movements is run, where the robot is interacted with during the task-contact. This is done to test the ability of the CEP,

detect a change of contact type or to detect the occurrence of a human interaction during task-contact. Figure 3.13 shows an example of control contacts during a task-contact (tool sliding on table depicted in Fig. 3.7(b)) as simulated with the Threshold-Based Method. The two task-contacts can clearly be distinguished in the sensor data. To interact with the robot during the task contact, the robot has been lifted from the table surface. These instances of control contact can be located in the external sensor force data. When the tool is lifted from the contact, the force in tool direction drops to around zero. Nevertheless, the Contact-Event detector indicates that a contact is still exerted. The Contact Type Classifier first classifies a control contact and then switches back to a classified task-contact. During the recording of this simulation, the system does not react to the contact event. The impedance control increases the force to counter the positional offset. Since higher forces are exerted on the operator, they can lead to a false classification. Therefore, only the beginning of the contact event is correctly classified. In a real scenario, as described in the previous section, the system would decrease the stiffness of the controller as a reaction to a control contact. Since the beginning of the control contacts are correctly classified, this would lead to a correct reaction during a real world task. In Fig. 3.14 the same recording is simulated with a Distance-Based Method. Here, only the interaction (control contact) is detected, for the system has knowledge about the task. Only small instances in the beginning of the task-contact are indicated as contact due to high impact forces on the external sensor and the inelastic impact on the table surface. Most of these false positives are filtered out by the CTC. The human interaction with the robot is indicated by the high values of the distance metric. The exerted contact is mainly classified correctly as control contact. The falsely classified collisions are again due to the constant stiffness of the impedance controller as explained above. If the stiffness is decreased, after the initial detection of a control contact, the high forces counter the interaction are avoided.

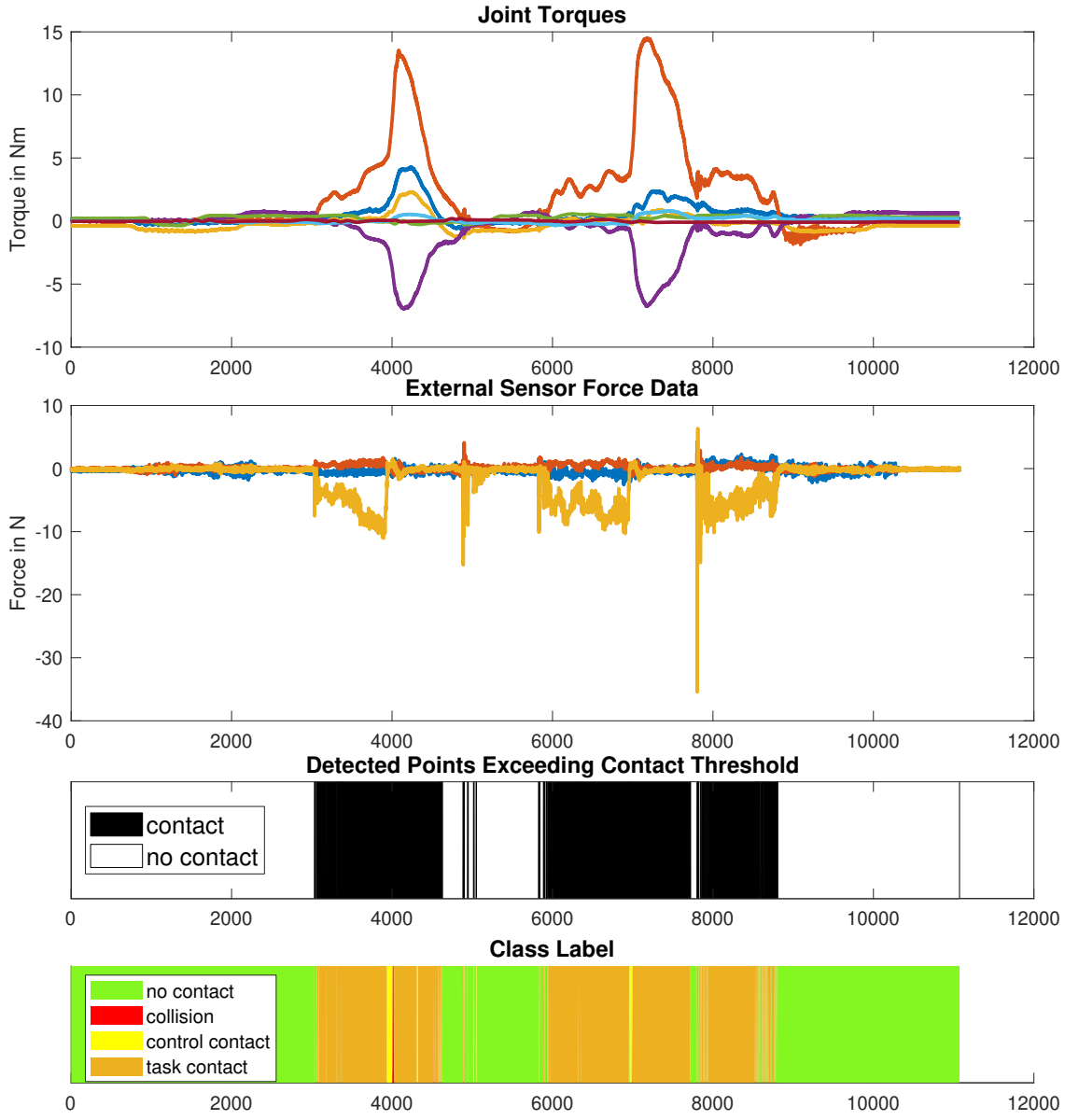


Figure 3.13: Simulation result of the CEP for TBM approach with two task-contacts from sliding on the table surface with two control contacts. The first two plots show the torque and force profile from the internal and external sensors. The last rows indicate points instances detected to be of a contact event as well as the predicted contact type.

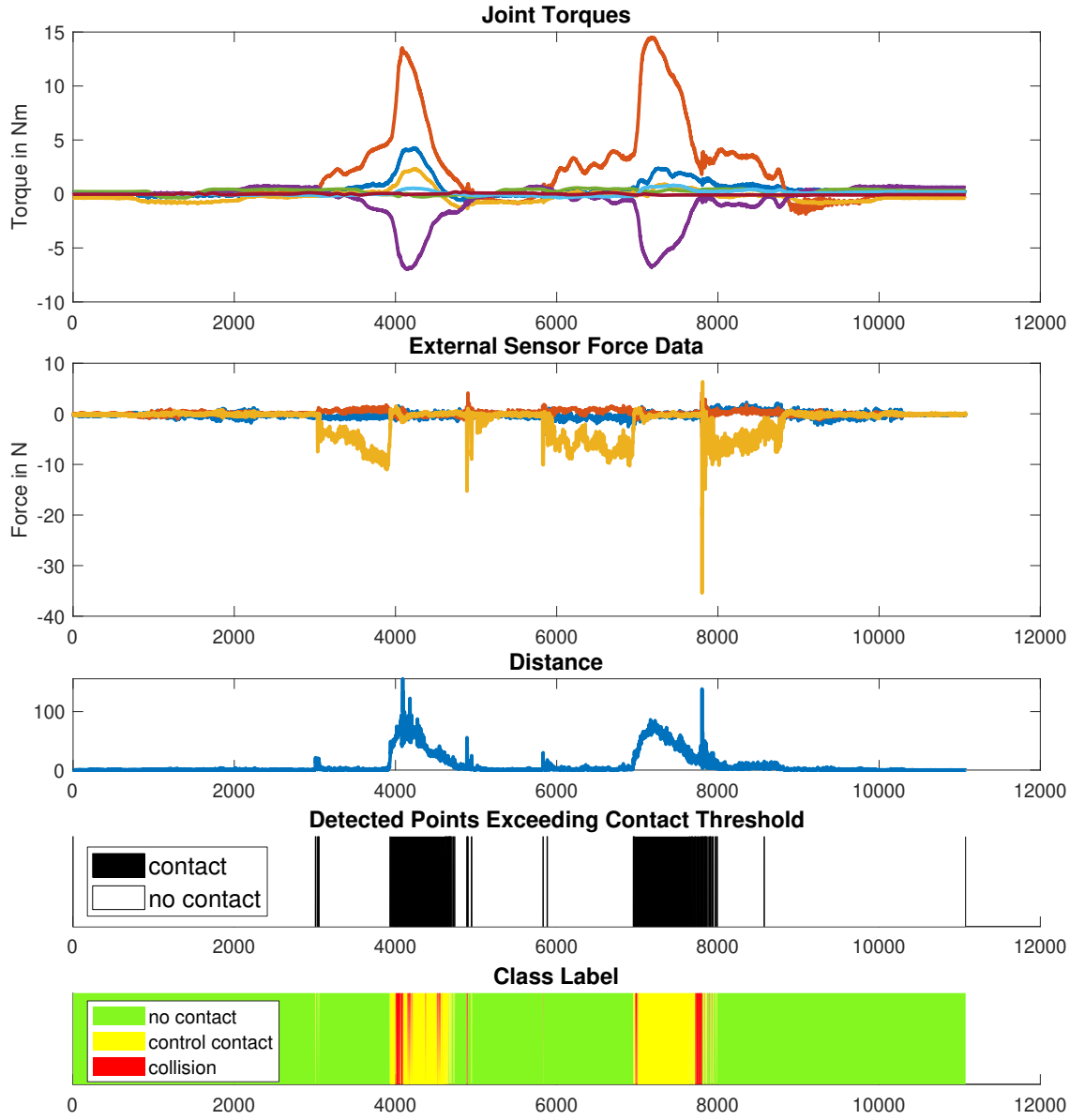


Figure 3.14: Simulation result of the CEP for DBM approach with two task-contacts from sliding on the table surface with two control contacts. The first two plots show the torque and force profile from the internal and external sensors. The third plot shows the distance offset from the reference sample. The last rows indicate instances detected to be of a contact event as well as the predicted contact type.

3.2 Experiment

The results of the simulation show that in principle both approaches for the Contact Event Pipeline work. Contacts resulting from the task as well as human interaction are detected and classified on prior recorded but real samples. It is therefore now the intention to experimentally evaluate the system. The goals of the experiment are the testing of the real time capability of the framework, the adaptability of the system to unknown settings and its performance on a real-world scenario. The setting of the experiment is chosen so that interactions with the robot can both be performed separately and during task contact. Therefore, a task must be found that includes long phases of continuous task-contact as well as movements in "free-air" that facilitates the mentioned contacts. The scenario chosen is a painting trail. Again, the experimental setup includes a KUKA LWR-IV+ robot with a wrist mounted jr3 force/torque-sensor and a Weiss Robotics WSG-Series gripper. To the end of the gripper a paint roller is fastened as depicted in Fig. 3.15.

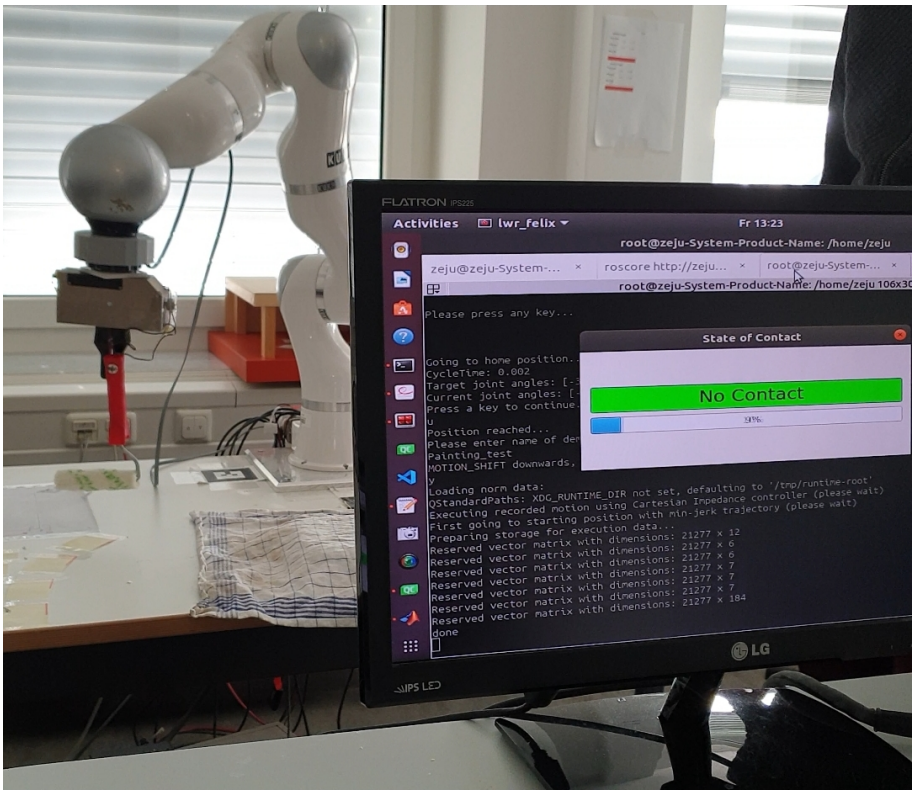


Figure 3.15: Experimental setup. A KUKA LWR-IV+ robot with a wrist mounted jr3 force/torque-sensor and a Weiss Robotics WSG-Series gripper is used. A paint roller is fastened to the end of the gripper. Two monitors are placed the line of sight of the user showing a dialog window indicating the progress of the task as well as the classified contact.

The executed task is new to the system and has not been part of the training or testing set. From its initial position, the robot moves towards the table until the paint roller touches the surface and applies small amounts of pressure. The contact to the surface is then kept and the robot performs a painting motion. Thereafter, the contact is lifted and the arm moves to a second position where the motion is repeated. After the second painting motion, the robot moves back towards its initial position. The user is informed of the current contact situation and type by a dialog window, indicating the type of contact as well as the progress of the current task as shown in Fig. 3.16.

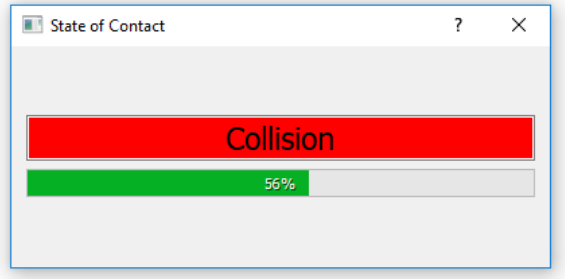
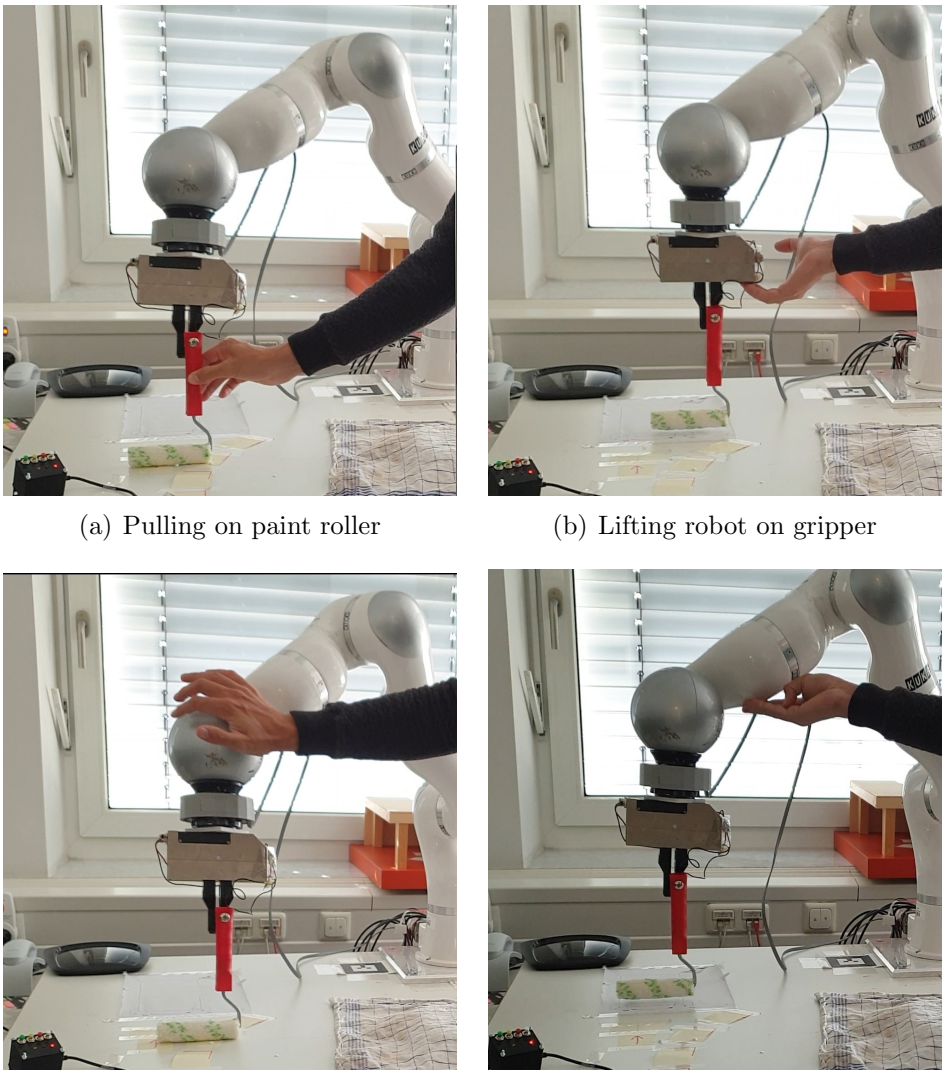


Figure 3.16: Contact dialog window indicating the current state of contact and the progress of the current task to the user.

The experiment is divided in two parts. In the first part of the experiment, to test the CEPs capability of distinguishing task-contact from normal ("free air") execution, the described motions are first run without any human interaction. The execution is monitored and the outputted contact types are analyzed. This test is run a total of twenty times for each approach. The second part of the experiment is performed to test the contact type prediction capability of the CEP. Another twenty trials are run for each approach and per contact type where interactions during execution are enabled. The contacts performed are of the remaining types (control contact and collision). They are performed both during task-contacts and in free air. The contacts are applied both on the robots body as well as on the tool and paint roller by pushing or pulling. Figure 3.17 shows different types of interactions with the robot. The contacts simulate practical interactions that could occur during the tested scenario. As illustrated in Fig. 3.17(b) and Fig. 3.17(d), the robot might be lifted of the ground to stop painting on a specific section. Or the tool might be pulled down to increase the contact force as in Fig. 3.17(a).



(a) Pulling on paint roller

(b) Lifting robot on gripper

(c) Simulated collision

(d) Lifting robot on arm

Figure 3.17: Four examples of interactions during the experiment to test the contact type prediction performance of the CEP.

3.3 Experimental Results

The first part of the experiment is conducted to analyze if the system can distinguish task contact from normal execution. The two approaches define two different methods of doing so. The Distance-Based Method uses the task knowledge encapsulated in the reference sample. Therefore, under this approach, task-contacts count as normal execution and should not be indicated to the observer. Consequentially, an indication of the remaining contact types (control contact or collision) is counted as a failed trial. Practically, this means that during the contact between the paint roller with the table surface, the dialog window should show "no contact". The CEP based on the TBM on the other hand is configured to detect and systematically classify task-contact. Here, twenty trials are recorded and the outputted classifications are shown in Fig. 3.18.

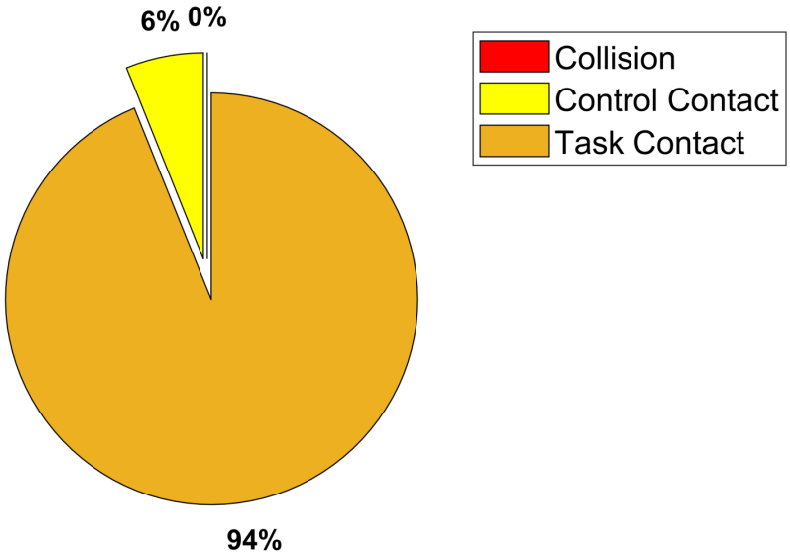


Figure 3.18: Classification output of all trial executions of pure task-contact as a pie chart using the TBM approach.

The trials for the CEP based on the DBM are all successful, i.e. no other contact has been shown during the trials. The classification labels from the TBM show a misclassification rate of six percent towards control contacts. One example of a complete experimental trial is depicted in Fig. 3.19. Two contacts, resulting from the painting motion, can clearly be distinguished in the sensor data. Regardless, the contact forces of the second painting motion are considerably lower. Therefore, the CED does not consistently detect the contact. The detected contacts are almost entirely classified correctly. Only a minor number of instances are classified as control contact around the mark of 0.5. Some of the contact instances of the second task-contact are falsely classified as "no contact" (false negatives). In general, for

both methods, the bare contact events have been correctly detected.

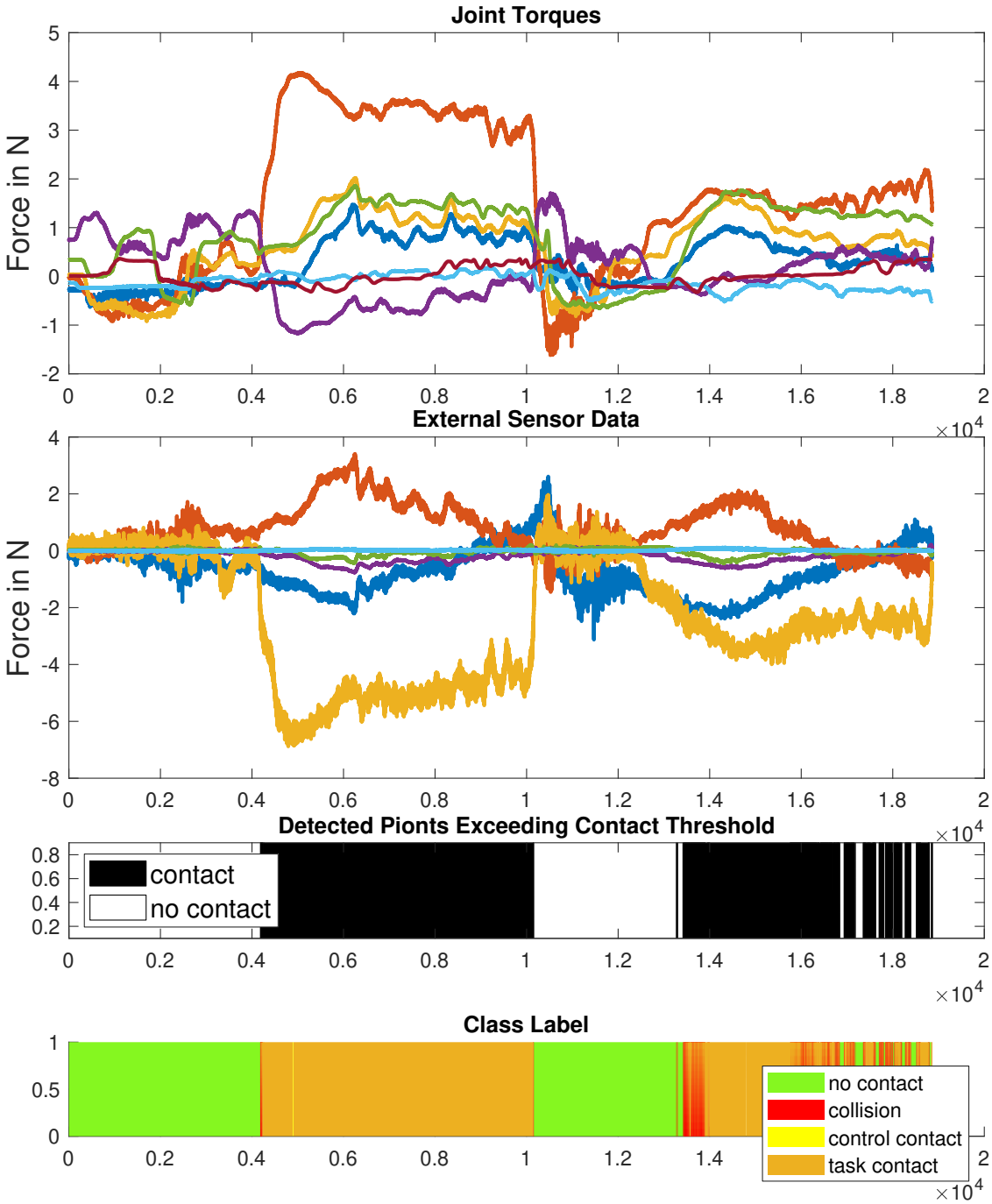


Figure 3.19: Classification output of one trial execution of pure task-contact using the TBM approach.

The goal of the second part of the experiment is to evaluate the performance of the CEP regarding human interaction. Since the setup is configured to come to an immediate stop after a collision, they can clearly be distinguished from control contacts during execution. First twenty control contacts are performed on the robot. A classified collision, consequentially leading to the termination of the motion, is counted as a failed trail. Likewise, when testing collisions, no termination of the motion and a classification as control contact is counted as a failed trial. Regarding the TBM, a classification as task-contact is counted as a fail as well.

		Type of performed contact	
		Control contact	Collision
Number of detected contacts	Correct	4	19
	Incorrect	16	1

(a) Total detected contacts TBM

		Type of performed contact	
		Control contact	Collision
Number of detected contacts	Correct	18	19
	Incorrect	2	1

(b) Total detected contacts DBM

Figure 3.20: Table showing total number of correctly and incorrectly classified contacts in the second part of the experiment.

Figure 3.20 shows the number of correctly and incorrectly detected contacts of the twenty trails for each contact. Using the TBM approach in Fig. 3.20(a), only four out of twenty control contacts are correctly classified while there is only one false classification on the collision side. From the confusion matrices in Fig. 3.21 representing the overall experimental classification results, it can be seen that all incorrectly classified control contacts are classified as task-contact (Fig. 3.21(a)). For the DBM, both eighteen and nineteen control contacts and collisions are respectively classified correctly (Fig. 3.20(b)). The confusion matrix in Fig. 3.21(b) depicts the total ratio of predicted contacts from the experiment for the second approach. Two examples of control contacts and a collision are depicted in Fig. 3.23 and Fig. 3.24 respectively. Figure 3.22 shows that the entire control contact is classified correctly, while in Fig. 3.24 the task is immediately stopped after the collision. Both the collision and the control contact occur during the task-contact.

		Predicted class		
		Task-contact	Control contact	Collision
True class	Task-contact	94%	6%	
	Control contact	80%	20%	
	Collision		5%	95%

(a) Confusion matrix from experiment for TBM

		Predicted class	
		Control contact	Collision
True class	Control contact	90%	10%
	Collision	5%	95%

(b) Confusion matrix from experiment for DBM

Figure 3.21: Confusion matrices for the results of the second part of the experiment.

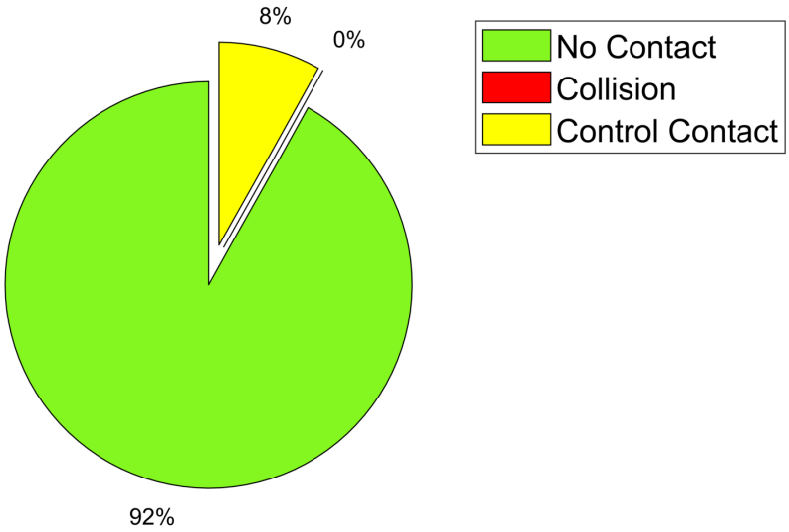


Figure 3.22: Classification output of a painting trial as pie chart using the DBM.

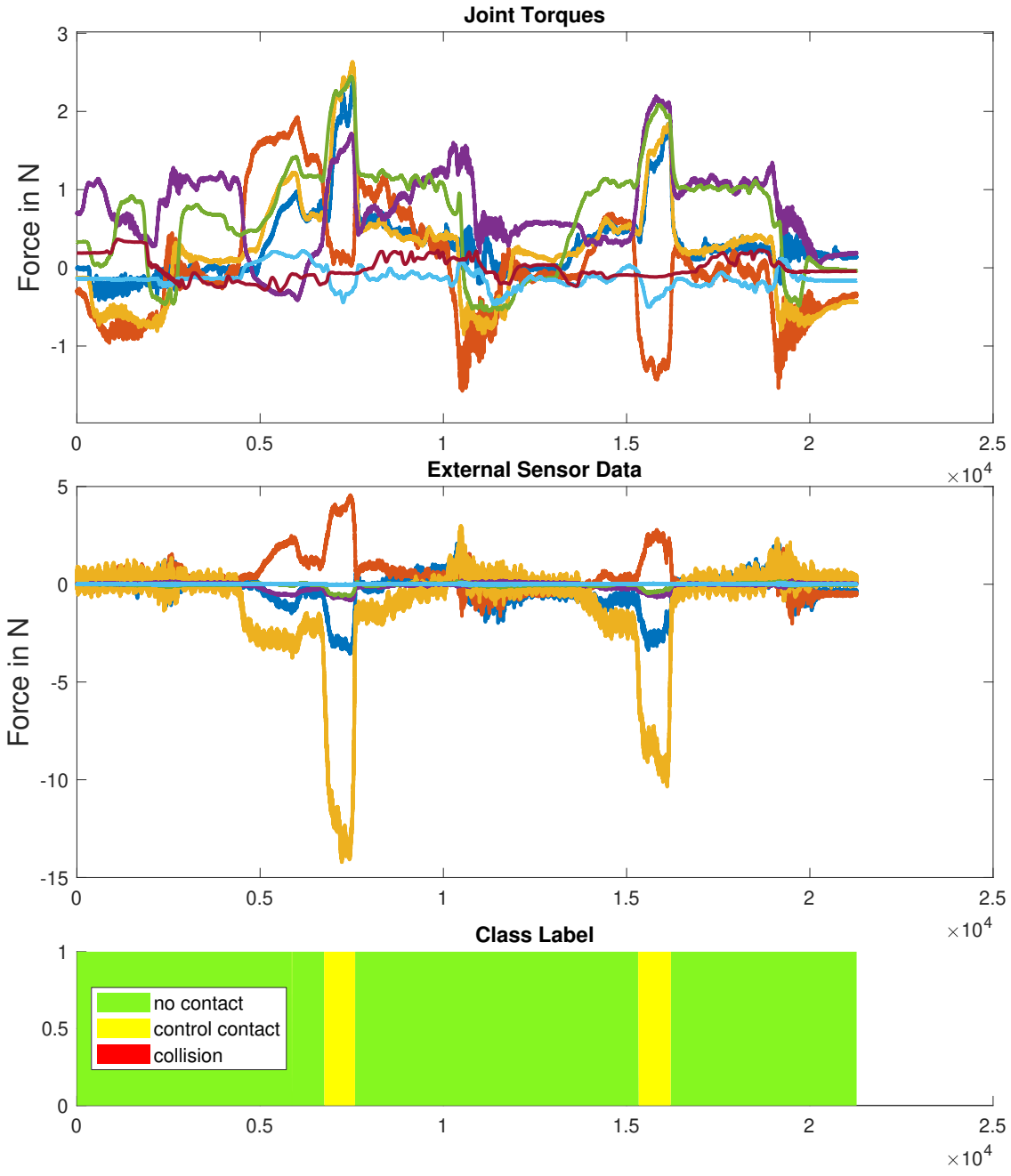


Figure 3.23: Classification output of a painting trial with two occurring control contacts using the DBM.

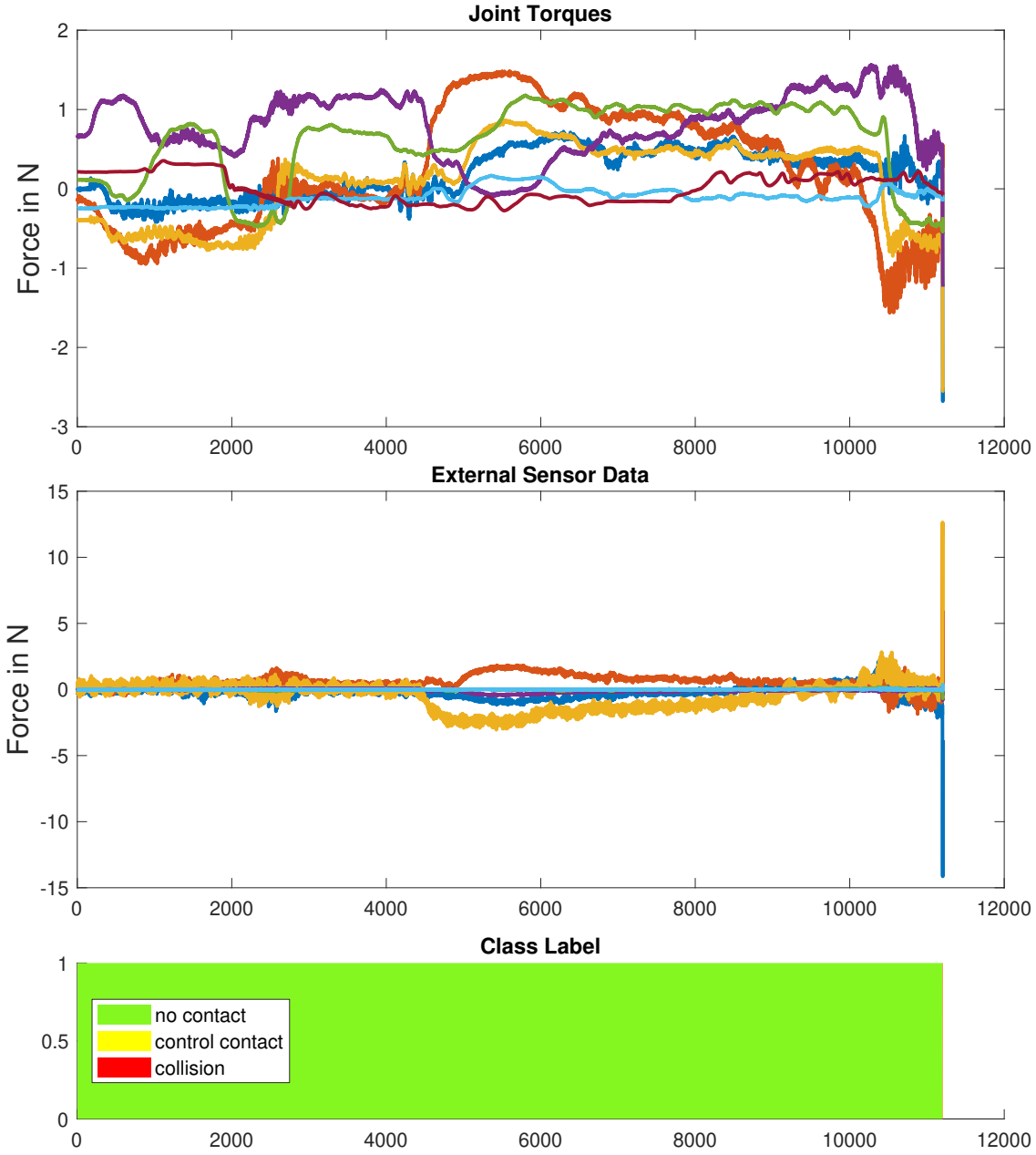


Figure 3.24: Classification output of a painting trial with a collision occurring using the DBM.

Chapter 4

Discussion

The performance of the Contact Event Pipeline, regarding its ability to detect and classify contact events, will be now discussed. The substantial differences between the Threshold-Based and the Distance-Based Method will be identified. Finally, the output of the Contact Type Classifier is qualitatively evaluated.

Regarding the reaction to a contact event, two methods are proposed, explained and tested. The main difference lies in the kind of data that is considered. While the TBM is directly applied to the sensor readings, the DBM uses an intermediate step of calculating a distance metric. Due to a non-existent ground truth, it cannot be determined from the simulation nor the experiment which of the proposed Contact Event Detectors works best. In theory, the DBM is more robust against perturbation from sensor noise, but both, simulation and experiment, show that both methods can accurately distinguish contact events from normal execution. Only the TBM is prone to errors for task-contacts with low exerted contact forces. Their sensor values fall below all the sensor thresholds and are therefore not detected. But this problem occurs mostly for the simulated task-contacts and not in the more realistic experimental task-contacts.

It is discussed in several parts of this work that reaction time is one of the most pressing factors for the real time applicability of the proposed pipeline. For the overall computational reaction time of the CEP, the most vital parts in terms of time consumption are separately analyzed. The results show that the overall time needed for the robot to be able to react is summed up by the delay of the sample window as well as the prediction. By using k-fold cross-validation, this delay is optimized to 0.17s. This value is confined by the cycle rate times the length of the sample window N and can only be reduced by a higher cycle rate. The other parts of the pipeline, the detection of a contact event and the calculation of the features, are quicker than the sampling time of 2ms. The delay from the prediction of 9.0 ms to 9.5 ms does not considerably prolong the reaction time. The resulting maximum computational reaction time of around 0.18 seconds is considerably short enough to

apply the process to a real robot. Compared to the state of the art, we achieved comparable reaction times with more extensive classification possibilities. Since the features are always computed and classified once a contact event is triggered by the CED, the reaction time can possibly be faster than computed upper bound.

With the optimized parameters a considerable good performance of the models in 10-fold cross validation is obtained. All the misclassification errors are well below one percent. This means that the chosen type of model is able to generalize well among new samples. This fact is further substantiated by the robustness, relevance and redundancy evaluation of the features. As shown by the results of the Minimum Redundancy Maximum Relevance algorithm in Fig. 3.5, all of the chosen features contribute toward the contact type classification. None of them is considerably more irrelevant than the other, which is shown by the small drops of the importance weights in between the predictors. Especially the newly added features i.e. work exerted by the joints and tool as well as the end-effector velocity and the Spectral Norm Derivative show great robustness between contact classes as seen in Fig. 3.6. Acceleration on the other hand is not as robust due to small value changes for the observations of different contact classes but is not less relevant for the classification according to the MRMR.

4.1 Simulation

The results from the simulation prove that the Contact Event Pipeline is able to detect contact events during task execution, filter false positives and correctly classify the indicated contact events for unknown recorded tasks. Both approaches, the distance-based and the threshold-based CED are able to precisely distinguish contact events from normal execution. Any false contact events triggered by high sensor noise, mostly from the external sensor, are correctly resolved by the Contact Event Classifier and eliminated. Only a minor portion of the correct contact events are falsely classified as noise and labeled as "no contact" like in Fig. 3.10 around the 3400 time stamp. This problem occurs entirely for the detection of task contacts for the TBM and is of a minor issue to the task outcome. An actual task-contact classified as a no contact is not desirable but does no harm. Solitary control contacts and collisions are not affected by this problem. The only further occurrence of this issue is during a task contact that is combined with an interaction. After the contact is lost, briefly no contact is detected both from the CED and consequentially from the CTC. This can be identified in Fig. 3.13 around the 8000 time stamp. Most of the contacts are classified correctly throughout their entire course. However, classified collisions tend to switch to task-contacts for the TBM approach and to control contacts for the DBM approach in the course of the contact. This has two specific reasons. First, the recording of the samples and secondly their labeling. When a

collision with the robot is recorded, the dummy is placed in the way of the trajectory as depicted in Fig. 3.2. This is done to simulate an operator interfering with the robot task during execution. The robot is instructed to move to a certain position and back to its original state several times. Since the dummy is placed at a specific position, the robot collides with it at what is an inelastic collision. It then further goes on to push against the dummy until the contact is finally lost and the robot moves back. This is not how a human operator would react. A human would move away from the robot after the initial contact (except if he is clamped by the environment). This is of course not possible with the dummy. Secondly, during a real task with a human operator, the robot would be instructed to stop or move back as discussed in Sec. 1.2. The second, more elastic part of the collision is therefore closer to a task-contact or a control contact sensor data wise. Considering these restrictions from the training sample, only the initial inelastic part of the collision is labeled accordingly. The rest of the contact is not labeled. When considering the possible reactions to a collision discussed in the prior chapters, only the initial part of the collision contact must be classified correctly, witch it is. If the robot is stopped due to a collision, the rest of the classification resulting from the simulation are therefore irrelevant and can be ignored. Since the initial part of the collisions are always classified correctly, the simulation can be regarded as successful. A small portion of control contacts are misclassified as task-contacts for the TBM. This usually occurs at the end of the interaction where the contact is lost. The features seem not as robust for this specific type of interaction between control and task-contact. This is not a problem for the DBM since no task-contact can be classified. The simulation results of the combined task- and control contact are also successful. Regarding the fact that a constant robot stiffness both complicates the human-robot interaction and consequentially the contact event classification, the misclassification during the human interaction can be justified. This problem does not occur during a real task, where the stiffness of the controller is adapted after the detection of a control contact.

4.2 Experiment

An experiment has been found that enables testing of all types of contacts. A specific focus is put on how the robot can be interacted with during and without task contact. It has been discussed in the several parts of this work that in order to facilitate intuitive interaction, it must be possible to interact with the robot on any part of its surface. Several different positions of interaction are tested during the experiment.

The results, especially for the second approach (DBM), show that both human interactions, collision and control contact, can be detected in all the tested locations. It can even be detected if a hand is put in front of the paint roller during the paint-

ing motion. Additionally, most of the performed interactions are correctly classified by the CTC. For classified collisions the task is immediately stopped. For control contacts the task is continued and the contact type is continuously indicated for as long as it is performed. It is notable that the evaluation metric of the experiment is different to the simulation. In the simulation, the performance of the classification for every instance is evaluated. For the experiment, the evaluation method is further aggravated. In an industrial setting, each contact event, resulting from a human-robot interaction, must be classified entirely correctly. Therefore, in this experiment, the classification of a contact event is evaluated as a whole. Both, the failed collision trials in Fig. 3.20(a) and Fig. 3.20(b) have resulted from simulated collisions that have been performed too lightly. One of the failed control contacts from the experiments using the DBM occurred after the robot has been put down too abruptly after being lifted.

For the TBM, it is difficult to entirely assure if all the control contacts on all possible robot and tool surface areas have been successfully detected as a contact event itself. Most of the performed control contacts have been classified as task-contacts. Therefore, if a control contact is performed during a task-contact and it is misclassified as such, it can not be said if it has been detected. But the results from the simulation show that it most certainly is. The classification of the control contacts itself does not meet the performance of the other contact types. The difference of these results compared to the simulation can be explained by the difference in the evaluation method.

From the output of the CTC, it can be clearly seen that the SVM performs well in predicting the correct contact type, considering the aggravated evaluation metric. In simulation, the initial contact instances are entirely classified correctly, as well as the further ongoing contact event. This accounts also for the experiment, except for control contacts using the first approach (TBM). The statistics summed up in the confusion matrixes in Fig. 3.21 underpin the good performance of the classifier. Only minimal errors occur during classification in simulation. This can also be recognized in the trajectories of Fig. 3.9 and 3.10 where the classification color changes during the contact event for a minor amount of time. The start of a contact event must be classified most reliably, especially for control contacts or collisions. For those contact types, the reaction must be correct from the start of the contact event. The performance of the classifier is also not restricted to the number of classes. Both the SVM model for the Threshold-Based Method and the SVM for the Distance-Based Method perform well with multiple contact classes. Only the hyperplane of the first model has been prone to error during the experiment.

Chapter 5

Conclusion

In this work, a novel Contact Event Pipeline for the discrimination of task-contact from human interaction is proposed. It is shown that the system is able to actively monitor the execution of learned tasks. More specifically, the set-up procedure lets the user easily record new tasks and monitor the execution without the need for time consuming parameter tuning. The proposed monitoring system can detect and classify contact events quickly, enabling a fast robot reaction. It is explained how the system is implemented using a Support Vector Machine for contact classification and two different methods for contact event detection. New features are incorporated alongside existing features from prior literature to classify task-contacts and human interaction. One of the proven approaches to distinguish and classify contact events, is able to actively classify task-contacts from the set of defined contact types. The other one uses specific task knowledge to distinguish and classify only human interaction from normal execution using a distance metric. Additionally, the trained model can filter out sensor noise from the detected contact events to avoid false positives. Both models require only the robots internal position/angle and force/torque data alongside an external wrist mounted force/torque sensor. Therefore, the proposed method can be used without requiring additional sensors like artificial skin or tracking devices. The internal and external sensor data are both gravitationally compensated and show only external contacts. The SVM models are trained on hand- and automatically labeled data consisting of example task- and human interaction contacts. The described system can be adapted to various robot types with the same sensor modality. Only the machine learning model must be adapted to the specific sensor data. The proposed features as well as the proposed Contact Event Detector and Contact Type Classifier are tested in simulation and on a real time experiment.

The results show that relevant and robot's features have been chosen to perform the required classification for the contact type prediction. With the features extracted from example contacts, the chosen model is able to perform even in unknown environments. Generally, it has been shown that the second approach, the DBM

with its task knowledge, performs better in distinguishing between task-contact and human interaction during the experiments. Especially the differentiation between task-contact and control contact is prone to error for the fist method. Due to a missing ground truth, it can be said that both detection methods are able to distinguish contacts among the normal execution. Again, the DBM based on the task knowledge has been less prone to sensor noise during the tests. What has been proven are the intuitive and safe interaction possibilities during task execution. Compared to the state of the art, this work includes task contacts into the system. It can widely distinguish between the contacts resulting from the task and human interaction. Furthermore, this solution is not bound to the initial contact, as in the related work, but expands over the whole duration of the contact. Therefore, enabling a wider range of possible reactions, a better adaptability to current industrial tasks and especially the completion of the desired task.

We started this work on the premise that there is an increasing demand for collaborative robotics in the industrial sector. We showed how the proposed system can be easily adapted to new robots and industrial settings. Additionally, the classifier has been able to predict well when used on unknown tasks. This serves the discussed need for flexibility in frequently changing industrial production lines. A robot integrated with our system is able to react appropriately to intuitive interaction even by untrained work personnel. Furthermore, we discussed that health and safety of the human operator or collaborative worker is one of the most fundamental concerns. With the rather strict evaluation metric, following industrial standards, for the real experiment, we have shown the classification and reaction capability of the Contact Event Pipeline with task knowledge incorporated in the second approach. The applicability of the proposed system for industrial purposes is therefore given.

Future work Task-contacts could also be indicated to the user within the DBM approach. This could be achieved by combining the two methods. A contact event as indicated by the threshold-based method would generally indicate a task-contact. If the distance-based method indicates a human interaction through analyzing the task-knowledge, this would overwrite the priorly indicated task-contact and could classify the interaction type. With this procedure, the user would gain all visual indication with both methods. Furthermore, to increase the robustness of the TBM to distinguish between task-contact and collision, their feature values must further be investigated. The model could be trained on more realistic samples of task-contacts from real industrial scenarios. Alternatively, the confidence of the prediction together with a confidence bound could be used. By incorporating a variable confidence band for the TBM, the minor problem of false positive contact events can further be reduced. This could also reduce the burden on the SVM model to filter contact events triggered by sensor noise. The threshold band can be multiplied by a factor e.g. derived from the exerted forces. It could be investigated if some of the

sensor dimensions contribute more towards the detection and classification of contact events. Lastly, a user study could be conducted with multiple probands, to further validify the experimental results and prove the intuitive interaction possibilities.

List of Figures

Acronyms and Notations

CED Contact Event Detector

CEP Collision Event Pipeline

CTC Contact Type Classifier

DBM Distance-Based Method

HRC Human-Robot Collaboration

HRI Human-Robot Interaction

IFR International Federation of Robotics

MRMR Minimum Redundancy Maximum Relevance

pHRI pysical Human-Robot Interaction

SVM Support Vector Machine

TBM Threshold-Based Method

Bibliography

[Alm18] Diogo Almeida. Tools for gravity compensation and sensor calibration for wrist-mounted force-torque sensors in robot manipulators., 2018. URL: `https://github.com/kth-ros-pkg/force_torque_tools`.

[BAA15] Tugce Balli Altuglu and Kerem Altun. Recognizing touch gestures for social human-robot interaction. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 407–413, New York, NY, USA, 2015. Association for Computing Machinery. `doi:10.1145/2818346.2830600`.

[BG12] Hetal Bhavsar and Amit Ganatra. A comparative study of training algorithms for supervised machine learning. *International Journal of Soft Computing and Engineering (IJSCE)*, 2, 01 2012.

[BT04] A. Bicchi and G. Tonietti. Fast and "soft-arm" tactics [robot arm design]. *IEEE Robotics Automation Magazine*, 11(2):22–33, 2004.

[BYK+02] Catherina Burghart, Sadi Yigit, Oliver Kerpa, Dirk Osswald, and Heinz Woern. Concept for human robot co-operation integrating artificial haptic perception. In *Intelligent Autonomous Systems*, volume 7, pages 38–45, 2002.

[CPC+16] Andrea Cherubini, Robin Passama, André Crosnier, Antoine Lasnier, and Philippe Fraisse. Collaborative manufacturing with physical human-robot interaction. *Robotics and Computer-Integrated Manufacturing*, 40:1 – 13, 2016. URL: `http://www.sciencedirect.com/science/article/pii/S0736584515301769`, `doi:https://doi.org/10.1016/j.rcim.2015.12.007`.

[DAHH06] A. De Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger. Collision detection and safe reaction with the dlr-iii lightweight manipulator arm. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1623–1630, 2006.

[DP05] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.

[EH02] Dirk M Ebert and Dominik D Henrich. Safe human-robot-cooperation: Image-based collision detection for industrial robots. In *IEEE/RSJ international conference on intelligent robots and systems*, volume 2, pages 1826–1831. IEEE, 2002.

[GOH15] S. Golz, C. Osendorfer, and S. Haddadin. Using tactile sensation for learning contact knowledge: Discriminate collision from physical interaction. pages 3788–3794, 2015.

[HADH08] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3356–3363, 2008.

[HASH09] Sami Haddadin, Alin Albu-Schäffer, and Gerd Hirzinger. Requirements for safe robots: Measurements, analysis and new insights. *The International Journal of Robotics Research*, 28(11-12):1507–1527, 2009. arXiv:https://doi.org/10.1177/0278364909343970, doi:10.1177/0278364909343970.

[HDA17] S. Haddadin, A. De Luca, and A. Albu-Schäffer. Robot collisions: A survey on detection, isolation, and identification. *IEEE Transactions on Robotics*, 33(6):1292–1312, 2017.

[HDOEW19] Khoi Hoang Dinh, Ozgur S. Oguz, Mariam Elsayed, and Dirk Wollherr. Adaptation and transfer of robot motion policies for close proximity human-robot interaction. *Frontiers in Robotics and AI*, 6:69, 2019. URL: https://www.frontiersin.org/article/10.3389/frobt.2019.00069, doi:10.3389/frobt.2019.00069.

[HHK+12] S. Haddadin, S. Haddadin, A. Khoury, T. Rokahr, S. Parusel, R. Burgkart, A. Bicchi, and A. Albu-Schäffer. A truly safely moving robot has to know what injury it may cause. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5406–5413, 2012.

[ifr18a] Demystifying collaborative industrial robots. 2018. URL: https://ifr.org/downloads/papers/IFR_Demystifying_Collaborative_Robots.pdf.

[ifr18b] Robots and the workplace of the futurerobots and the workplace of the future. 2018. URL: https://ifr.org/downloads/papers/IFR_Robots_and_the_Workplace_of_the_Future_Positioning_Paper.pdfhttps://ifr.org/downloads/papers/IFR_Robots_and_the_Workplace_of_the_Future_Positioning_Paper.pdf.

[IIN03] Koji Ikuta, Hideki Ishii, and Makoto Nokata. Safety evaluation method of design and control for human-care robots. *The International Journal of Robotics Research*, 22(5):281–297, 2003. arXiv:https://doi.org/10.1177/0278364903022005001, doi:10.1177/0278364903022005001.

[KDA18] A. Kouris, F. Dimeas, and N. Aspragathos. A frequency domain approach for contact type distinction in huma-robot collaboration. *IEEE Robotics and Automation Letters*, 3(2):720–727, 2018.

[KFE12] R. Killick, P. Fearnhead, and I. A. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598, 2012. arXiv:https://doi.org/10.1080/01621459.2012.737745, doi:10.1080/01621459.2012.737745.

[KKW07] D. Kubus, T. Kroger, and F. M. Wahl. On-line rigid object recognition and pose estimation based on inertial parameters. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1402–1408, 2007.

[KLC15] Mohsen Kaboli, Alex Long, and Gordon Cheng. Humanoids learn touch modalities identification via multi-modal robotic skin and robust tactile descriptors. *Advanced Robotics*, 29(21):1411–1425, 2015. arXiv:https://doi.org/10.1080/01691864.2015.1095652, doi:10.1080/01691864.2015.1095652.

[KTFM18] Johan Kildal, Alberto Tellaeche, Izaskun Fernández, and Iñaki Maurtua. Potential users' key concerns and expectations for the adoption of cobots. *Procedia CIRP*, 72:21 – 26, 2018. 51st CIRP Conference on Manufacturing Systems. URL: http://www.sciencedirect.com/science/article/pii/S2212827118302592, doi:https://doi.org/10.1016/j.procir.2018.03.104.

[Lav05] Marc Lavielle. Using penalized contrasts for the change-point problem. *Signal Processing*, 85(8):1501 – 1510, 2005. URL: http://www.sciencedirect.com/science/article/pii/S0165168405000381, doi:https://doi.org/10.1016/j.sigpro.2005.01.012.

[MMP+10] G. Michalos, S. Makris, N. Papakostas, D. Mourtzis, and G. Chryssolouris. Automotive assembly technologies review: challenges and outlook for a flexible and adaptive approach. *CIRP Journal of Manufacturing Science and Technology*, 2(2):81 – 91, 2010. URL: http://www.sciencedirect.com/science/article/pii/S1755581709000467, doi:https://doi.org/10.1016/j.cirpj.2009.12.001.

[PEBK16] D. Park, Z. Erickson, T. Bhattacharjee, and C. C. Kemp. Multimodal execution monitoring for anomaly detection during robot manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 407–414, 2016.

[PHSA11] J. Park, S. Haddadin, J. Song, and A. Albu-Schäffer. Designing optimally safe robot surface properties for minimizing the stress characteristics of human-robot collisions. In *2011 IEEE International Conference on Robotics and Automation*, pages 5413–5420, 2011.

[RBM+10] A. Rodriguez, D. Bourne, M. Mason, G. F. Rossano, and J. Wang. Failure detection in assembly: Force signature analysis. In *2010 IEEE International Conference on Automation Science and Engineering*, pages 210–215, 2010.

[RŠK03] Marko Robnik-Šikonja and Igor Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Machine learning*, 53(1-2):23–69, 2003.

[YHH+97] Y. Yamada, Y. Hirasawa, S. Huang, Y. Umetani, and K. Suita. Human-robot contact in the safeguarding space. *IEEE/ASME Transactions on Mechatronics*, 2(4):230–236, 1997.

[ZRKS04] Michael Zinn, Bernard Roth, Oussama Khatib, and J. Kenneth Salisbury. A new actuation approach for human friendly robot design. *The International Journal of Robotics Research*, 23(4-5):379–398, 2004. arXiv:https://doi.org/10.1177/0278364904042193, doi:10.1177/0278364904042193.

License