Multi-Scenario Contacts Handling for Collaborative Robots Applications

Dmitry Popov, Stanislav Mikhel, Rauf Yagfarov, Alexandr Klimchik, Anatol Pashkevich

Multi-Scenario Contacts Handling for Collaborative Robots Applications

Dmitry Popov[1,2], Stanislav Mikhel[1], Rauf Yagfarov[1], Alexandr Klimchik[1] and Anatol Pashkevich[2]

***Abstract*— The goal of this work is to propose a way of dealing with physical interactions for collaborative robots that will ensure the safety of a human operator and improve the performance of a common task by implementing multiple robot behavior scenarios. In this scope, all collisions of a robotic arm are detected and analyzed to chooses an appropriate reaction strategy. The points of contact on the robot's surface for each collision are estimated, the external forces are identified and collisions are classified by the set of predefined categories. Based on these categories and the current robot state, the algorithm chose an appropriate behavior scenario.**

All presented algorithms are based only on proprioceptive sensors information and were tested in a simulated environment and on the real collaborative robots KUKA iiwa and Universal Robots UR10e. The result for contact localization showed 4 cm mean accuracy, the classification algorithm was able to identify collisions with hard and soft objects with 98% accuracy for KUKA iiwa 14.

I. Introduction

Human-robot collaboration is most important in production lines with large variety of products when the production operations require human-specific agility and skills. In this case a robot and a human work on the same task in the shared environment. By combining human intelligence and agility with robot strength, accuracy and robustness it is possible to improve the performance of this work cell. On the one hand, the robot will help the human with physically demanding tasks, on the other hand, the robot will have to work in a complex dynamic environment. This will increase requirements for robot control to predict human intentions, actions, and provide necessary assistance. During this interaction physical contacts between a robot and a human are unavoidable and should be taken into consideration [1].

Human safety is the number one priority in human-robot collaboration and that is why the majority of existing solutions are mainly concentrated around the safety aspect where the robot slows down, stops in the presence of a human, or turns on compliance mode. But physical contact is not only a negative event, it could be used to improve performance by enabling safe physical interaction. Besides, physical interaction is one of the simplest and most intuitive interaction methods even between humans, so it allows us to enable additional control or communication channel with a robot. To achieve this goal, the robot needs to properly handle all collision events which could include contact with a human, environment, another robot, workpiece, etc.

In this work, the methodology for multi-scenario contact handling is proposed. It consists of several steps combined into one pipeline and allows to easily implement safe and complex collaboration scenarios. These steps are: (i) collision detection is done by external torque estimation; (ii) collision localization, where the exact point of contact in the global or local coordinate frame is estimated; (iii) contact event classification using the set of predefined characteristics; (iv) information about the collision class and location is analyzed to activate one of the reaction policies.

The contributions of our work are the following:

1) We introduced the methodology to handle a physical interaction between robot-human/obstacle/task by utilizing different scenarios and dynamical switching between them.
2) The proposed approach covers all stages of a contact: from initial detection to classification and robot reaction to it. Most of the existing works consider only one aspect of interaction: contact detection, localization or classification.
3) We presented a modified version of the contact particle filter for contact localization and compared its performance with existing methods in simulation and on the real KUKA iiwa 14 robot.
4) We proposed a classification topology, which distinguishes contacts with soft and hard objects based on a deep neural network.
5) We demonstrated the reaction module, implemented as a finite state machine. This module chose an appropriate reaction according to the current state of the robot and the type of collision.

Together the developed algorithms could not only improve the safety of a human in a robot workspace but also increase the performance of the work cell by activating different scenarios of collaboration.

II. Related Work

Each step in multi-scenario contact handling pipeline is a challenging task itself. Here we give a brief overview of existing works on four main steps of a contact processing: Detection, Localization, Classification and Reaction.

A. Detection

The collision detection gives a binary answer to the question "does the robot have a collision with an external object?". Contact points can be located anywhere on the

[1] Center for Technologies in Robotics and Mechatronics Components, Innopolis University, Innopolis, Russia {d.popov, s.mikhel, r.yagfarov, a.klimchik}@innopolis.ru

[2] IMT Atlantique Nantes, Nantes, France and Laboratoire des Sciences du Numérique de Nantes (LS2N), Nantes, France anatol.pashkevich@imt-atlantique.fr

robot surface, from the base to the end-effector or tool. The main demands for this step are response time (from the moment when collision event happens to the moment when software sets the collision flag), which should be as short as possible, and detection precision that minimizes the number of false-positive classifications. Since it is a binary problem, most of the methods for collision detection use comparison of the input signals with some threshold. The simplest way is to monitor changes in motor current or torque values [2], [3]. These methods do not require a robot model. Another approach is based on the nominal robot model and comparing the model to the actual joint torques. This technique is similar to direct torque estimation or inverse dynamics approach [4]. More complicated algorithms allow not only to detect a collision but also to give an estimation of external torque (external torque here is referring to the part of the joint torque which was caused by the external force in the contact event), they are based on disturbance observers.

Currently, the momentum observer proposed by De Luca [5] is one of the most popular solutions for estimation of external load, not only for robotics arm but also for other kinds of robotic systems like drones [6]. The momentum observer does not require inertia matrix inversion and joint acceleration values, but at the same time provides a first-order filtered version of the external torques and, as a result, has asymptotic convergence for the constant external load only. A survey on the detection and estimation methods is presented in [7]. Collision detection based on deep neural networks was proposed in [8], which showed a reliable detection for cyclic operations.

B. Localization

When the fact of the robot collision with an external object is detected, the exact location of the collision area should be identified. Here we consider methods based on proprioceptive sensors only, in most cases, these are 1D joint torque sensors. Knowledge about the collision point is important in the scope of choosing the right reaction policy later. One of the first works in this area considered a planar 3 DoF robot and identified a collision link surface [9]. The result of their algorithm depended on simplified robot shape and observed disturbance torques. Buondonno and De Luca in [10] studied external force isolation using a force sensor mounted at the robot base, which gave a reliable solution at any point of the robot no matter how close contact is to the base.

In [11] the authors analyzed the localization of the collision point placed on the internal axis of the robot links. Such a method reduced the dimensionality of the problem and allowed to use global optimization by only two constrained parameters. Later this method was used in [12], [13] and compared with machine learning approaches.

Manuelli and Tedrake [14] suggested using a particle filter for collision localization. In this method, each particle includes information about its location on the surface of the link, and weight is considered as a probability of contact at this point. Probability is estimated with the assumption that external force is applied in the friction cone. After this publication, several other research groups investigated similar methods. In [15] the motion model of particles was modified with a combination of small self-movements of the robot to increase accuracy. Monte Carlo localization was used in [16] and compared to direct optimization and machine learning methods. In both works, the authors examined forces acting to the normal of the robot surface constrained with a friction cone, which allowed to create more particles and decrease the computational load. In one of our previous works [17] the contact isolation was done using two-step optimization on the mesh surface of the robot KUKA iiwa 14.

It should be noted that collision localization can be also done using external sensors. Tactile artificial skin makes the problem trivial [18], [19] and solves detection and localization tasks at the same time, but it is expensive and requires additional calibration for the robot. Vision sensors such as a camera or RGB-D sensor Kinect [20], [21] on the one hand, can also help, especially in cases of multiple collisions and safe human-robot coexistence, where robot can avoid unnecessary collisions [22]. On the other hand, vision systems depend on the field of view and lighting conditions.

C. Classification and Reaction

By evaluating the characteristics of the collision it is possible to attribute it to one of the predefined classes. This allows us to understand the context of the contact event and make a decision for the reaction. In [23], [24] contacts with a human are divided into intentional and unintentional. In another work contacts are classified as collision with a soft object (human) and a hard (workpiece, another robot, walls) object, and duration was either single or continuous [12]. Parusel at el. [25] designed a control architecture for realizing human-friendly behaviors and intuitive state-based programming with four control modes. The first mode is autonomous in case of human absence, the second mode is compliance with human presence, the third is a collaborative mode with a human in the loop and the last mode is the case of a fault. Magrini and De Luca in their work [26] implemented a finite state machine with 3 basic robot states: idle state, null space redundancy state and high compliance mode. In our previous works [27], [28] a similar finite state machine was proposed for switching between six different scenarios of robot behavior.

III. Methods

Now we describe the implemented methods that were used in multi-scenario contacts handling algorithm. The detection part relies on two already well-known approaches: momentum observer [5] and disturbance observer [29]. In the localization part, we propose to use a modified contact particle filter [14], in contrast to which we change a robot surface representation and the motion model to reduce computation time. For the classification, the deep neural network is used to classify ”hard” and ”soft” collisions. The finite state machine is used for the reaction part of our pipeline. Here states of finite state machine represent all possible

reactions, transitions between these states are defined by the collision events and their characteristics.

A. Detection Algorithms

Different techniques allow to identify collisions based on proprioceptive sensors only, without additional external sensors. The general idea behind them is to compare measured data with the predicted robot state. The predicted state relies on the robot dynamic model. Thus, if the dynamic properties of the robot are identified then the prediction of the current parameters could be obtained.

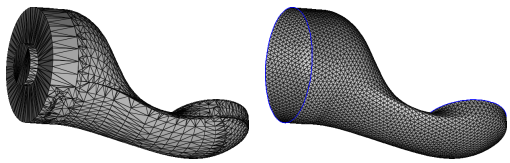
The disturbance observation technique is usually based on the assumption that the perturbation rate of some parameter is proportional to the current difference between the perturbed and estimated values. It allows us to estimate the disturbance by integration. The parameters can be torques, moments, velocities, or some others depending on the observer algorithm. The choice is usually determined by the desire to simplify calculation or reduce errors. In particular, it is the common approach to exclude joint acceleration because its calculation is based on numerical differentiation and decreases accuracy.

Three external torque observers were implemented. The first one is a momentum observer [5]. Authors use rate of change in momentum in order to estimate residual torque. Authors of the second method, disturbance observer [29], eliminate acceleration from equations with the help of special matrix $\boldsymbol{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ are the vectors of joint angles and velocities. The third observer is a modification of the momentum observer with a sliding mode technique that should deal with a wider class of disturbances [30].

B. Localization Algorithm

Robot Surface Representation: Although the surface of the robot is often approximated with primitive shapes like cylinders, for the collision isolation task it will produce a poor result, especially with a robot of a complex shape like KUKA iiwa. To overcome this, we propose to use a modified and extracted surface of the robot links from its .STL model. The robot .STL models are widely used for visualization purposes, but their accuracy is well enough for the localization task. The initial model, imported from the .STL file, presented in Fig. 1a. It has an internal area of the link and some surfaces, unavailable during physical interaction with a robot. Also, it is more convenient and computationally easier to work with a finite number of points, sampled on the robot surface, than to find a point on the facets of the link. Of course, the points should be equally distributed and distance between these sampled points should be less than desired accuracy. The modified mesh of the robot link is shown in Fig. 1b and obtained by removing unreachable regions and isotropically resampling mesh.

The remeshed link could be represented in the form of the graph where the vertex is a node ($\boldsymbol{N}$) of the graph and faces are edges ($\boldsymbol{E}$) as $\boldsymbol{G} = (\boldsymbol{N}, \boldsymbol{E})$. Each node $\boldsymbol{N}_i = \langle \boldsymbol{p}_i, \boldsymbol{n}_i, \boldsymbol{nb}_c \rangle$ includes information about its position in a joint coordinate frame $\boldsymbol{p}_i$, $\boldsymbol{n}_c$ is a normal vector in this point and a list of neighbors $\boldsymbol{nb}_c$. This graph is obtained offline and will be constant during the run-time.



(a) Imported .stl file (b) Modified mesh

Fig. 1. KUKA iiwa link mesh representation. In (a) mesh has unreachable for contact areas. In mesh (b) those areas were removed and the mesh surface was isotropically resampled.

Contact Particle Filter on Graph: The localization of the contact point and estimation of external force formulated as a nonlinear optimization problem

$$\min_{F_p, \boldsymbol{p}} \left\| \boldsymbol{r} - \boldsymbol{J}_p^T \boldsymbol{F}_p \right\| \tag{1}$$

where $\boldsymbol{J}_p$ is a Jacobian for a collision point $\boldsymbol{p}$, $\boldsymbol{r}$ is an estimation of external torque, $\boldsymbol{F}_p$ is an external force, applied to the contact point.

It should be noted that here we assume only external forces, without external torques. The use of only force vector applied to the robot surface will make identification possible for links after 3rd in the case of 1D joint torque sensors. To make a result of this optimization more physically correct, additional constraints are needed.

The first constraint assumes that contact point $\boldsymbol{p} \in \boldsymbol{C}$, where $\boldsymbol{C}$ is a surface of the robot.

The second assumption is that $\boldsymbol{F}_p$ lies inside a friction cone with friction coefficient μ, defined by the normal vector in point $\boldsymbol{p}$. This constraint could be approximated with a polyhedron for computational advantages:

$$\boldsymbol{F} = \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{f}_{fc,i}, \ \min_{\boldsymbol{a} \geq 0} \left\| \boldsymbol{r} - \boldsymbol{J}_{fc,a}^T \boldsymbol{a} \right\| \tag{2}$$

Here $\boldsymbol{J}_{fc,a} = \boldsymbol{J}_{fc} \begin{bmatrix} \boldsymbol{f}_{fc1} & \dots & \boldsymbol{f}_{fcn} \end{bmatrix}$ is Jacobian $m \times n$ for each support vector $\boldsymbol{f}_{fc}$ in contact point, m is a number of joints, n is an approximation degree, $\boldsymbol{a}$ is a support vector weights $n \times 1$. Since $\boldsymbol{r}^T \boldsymbol{r}$ is a constant, and friction cone is a convex set, it is sufficient to formulate it as a quadratic program:

$$QP = \min_{\boldsymbol{a} \geq 0} \left\| \boldsymbol{a}^T \boldsymbol{H} \boldsymbol{a} + \boldsymbol{g}^T \boldsymbol{a} \right\| \tag{3}$$

where $\boldsymbol{H} = \boldsymbol{J}_{fc,a} \boldsymbol{J}_{fc,a}^T$ is a square of Jacobian, $\boldsymbol{g} = -2 \boldsymbol{J}_{fc,a} \boldsymbol{r}$.

The minimal value of QP is zero, which corresponds to complete coincidence, and the maximum value depends on the model error.

The exact collision location is estimated using particle filter [14], which in contrast to the original work operates on the robot surface graph $\boldsymbol{G}$. This allows avoiding the computational complexity of a motion model update, where

new positions of the particles are projected on the surface of the robot by finding ray-mesh intersections.

To describe the proposed particle filter motion and measurement model are presented.

Motion Model: The motion model is used for updating the particle location. Due to the unpredictable behavior of the interaction point on the robot surface, it is not possible to define a model with good generalization. For example, assuming contact point motion on the surface will have good results when a human grabs the robot. At the same time, robot collision with a wall will benefit from a model with a contact point in the base frame. For our particle filter, we use a random walk policy as the motion model, which will generate reasonable results for most cases.

The random walk policy on the graph means choosing a random neighbor node to the current node iteratively. The number of iterations is defined by a random integer between 0 to max_steps variable. This variable depends on the weight of the current particle, so the particle with a high weight will have a lower maximum walk distance.

Measurement Model: The measurement step applies weights to the particles. The weight of each particle is equal to the residual between the measured external torque and recreated torque in this particle. For further analysis, it is useful to map the result into the interval from 0 to 1:

$$w \sim exp(-\alpha\, QP) \tag{4}$$

where α is scaling coefficient, normalization constant is omitted here.

Multi-collision Cases: To extend the approach to a multiple collision identification, an additional particle set should be added similarly as it was described in [14]. In this case, each particle set will have its own solution with a contact point, force in this point, and a generated torque. The sum of generated torques by all particle sets will be equal to total external torque.

Unfortunately for collaborative robots with a 1D torque sensor per joint, it's hard to detect more than 2 collisions in practice. The first problem is that the resulting torque from all collisions has no unique solution and could be described as a single collision. It can be partly avoided by assuming that collisions happen sequentially. The second problem is an amplitude of collision impact: with a large difference in amplitudes, one of the collisions could be identified as noise for a significantly larger second collision after torque normalization. To improve identifiability, we made an additional assumption that one of the collisions is placed at the end-effector of the robot. Here end-effector payload is not modeled in the robot dynamics. If a robot performing a contact operation like pick and place, drilling, and so on, there is one collision associated with a task at the end-effector and a second collision in the intermediate point of the robot in case of a collision with the environment/human is possible. This leads to set up with particle filter with one set and one fixed particle from a second set for representing collision in intermediate point and end-effector of the robot.

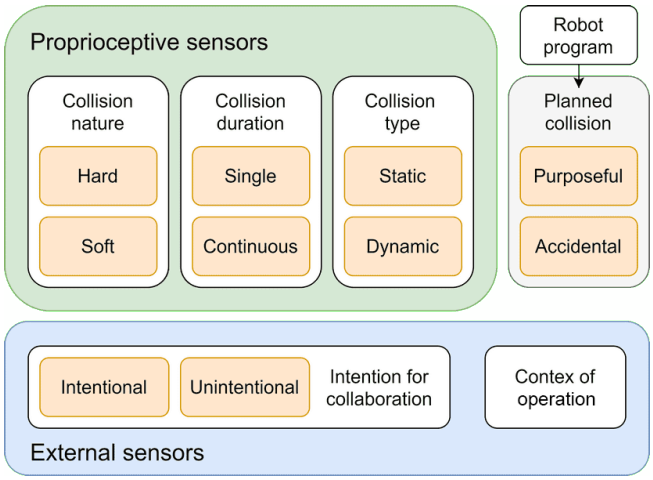


Fig. 2. Collision classes. Collision duration, nature, and type classes could be obtained using only internal sensors of the robot. By using an external vision system it is also possible to extract the intention and context of interaction. The possibility of a planned contact could be predefined for each step of the robot program.

C. Collision Classification

Depending on the context of a collision event, the outcome will be different. Here we try to understand the context of a collision based only on the robot's internal sensors. The classification starts when a collision is detected. Currently, we defined 6 criteria for the collision event, but it could be easily extended. The classification diagram is presented in Fig. 2.

The first criterion is the nature of a colliding object, we divide all possible variants into hard objects (could be a wall, workpiece, etc.) and soft objects (human in most cases). The object nature could be classified by analyzing the time series of joints torques and positions. Since we use only the internal sensors of the robot, it is not possible to reliably distinguish contact with humans and contact with cushioned furniture, for example. Thus, we have to assume all soft contact as contacts with a human. In the case of industrial applications, soft and hard contacts usually correspond to the operator and the robot environment. Consequently, if a collision happens with a human, we can define if it was intentional as a part of a human-robot collaboration or the collision was incidental.

Every collision could be purposeful or accidental. The collision is marked as purposeful if it is expected at the current stage of the robot program, otherwise, it is accidental. If the process is non-contact, the collision in most cases would be accidental, but if we expect collaborative work, conditions of the purposeful interaction could be defined.

The duration of the contact event defines if it is single or continuous. When the contact time is longer than a certain threshold value, the collision is continuous, otherwise, it is single and can be caused by an accidental human touch, for example.

By the contact location extracted from the localization algorithm, we can mark this collision as static or dynamic.

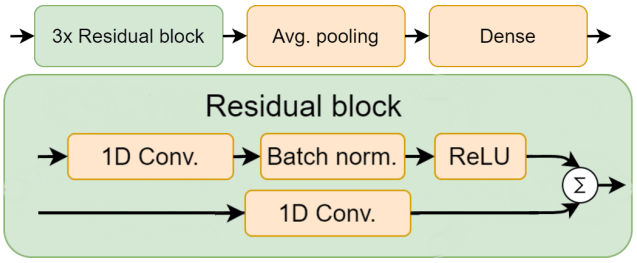


Fig. 3. Structure of proposed DNN for soft/hard collision classification. As input, a fixed time window for the contact external torques is used. The class of the collision is output.

Depending on the chosen coordinate frame, collision is static when coordinates of the contact point are fixed with relation to frame and dynamic if coordinates are changing in time. The idea behind that is to separate dynamic objects such as a human or another robot with static immovable objects like walls in the robot workspace.

The accidental/purposeful class is given on the stage of robot programming and depends on a specific task. Instead, single/continuous, soft/hard, and static/dynamic classes are estimated online from the properties of the collision. Particularly the soft/hard class identification done by the deep neural network is shown in Fig. 3.

The input for the network consists of external torque values for a fixed time window. In the case of multiple collisions, only part of the total torque that was generated by this collision is used. The use of velocities in the case of a single collision could help with classification, but in practice, it decreases the accuracy of the model when used for a second collision. The output of our network is a class of collision: soft or hard. The classification works only if at least one collision is detected.

A deep neural network for collision classification consists of 3 identical sequential Residual blocks. Each Residual block has two paths for input data. The first path consists of sequential 1D convolution, batch normalization, and ReLU activation layers. The group of these three blocks is repeated 3 times and connected sequentially. The second path includes one 1D convolution layer. Next, both paths are connected using the addition operation. After 3 Residual blocks, 1D global average pooling is performed and then the network has the last feedforward layer with a softmax activation function.

Data from proprioceptive sensors is not enough to estimate the context and intention of the collision events. This implies the use of external sensors like a camera or RGB-D with classification algorithms based on deep convolutional networks, image processing, and that is out of the scope of this work.

D. Reactions

The main goal for the collaborative robot is to be safe for humans, the environment, and themselves, and simultaneously execute the task most efficiently. To achieve this goal the robot should have a strategy for every possible collision

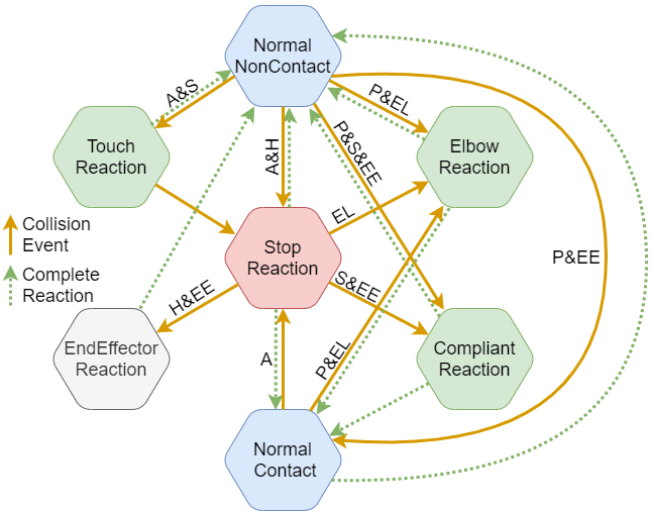


Fig. 4. Proposed finite state machine with states and transitions. Collision by location: in end-effector ($\boldsymbol{EE}$), in elbow ($\boldsymbol{EL}$); by nature: soft ($\boldsymbol{S}$), hard ($\boldsymbol{H}$); by current task: purposeful ($\boldsymbol{P}$), accidental ($\boldsymbol{A}$).

and a switch that will turn this strategy on and off. Here, we propose to use a finite state machine.

In the scope of our methodology, states of the finite state machine are reaction strategies, contact events are transitions and contact properties are transition conditions. The end-user can define states and transitions between them according to the desired robot behavior. Each state includes several robot actions, each transition has a condition on collision classes.

The following example shows the possibilities of our approach. Consider a collaborative work cell, where a robot has the task of tightening screws. The end-effector of the robot is equipped with a camera for screw detection and a gripper tool. The finite state machine for this example is presented in Fig. 4. The robot has 2 main tasks. The first task executes non-contact operation (“Normal NonContact“ state), for example, visual detection of screws in a workpiece. The second task is contact operation (“Normal Contact“ state), for example tightening of a screw with a tool.

The basic reaction for all accidental collisions (”A”) is defined by the safety policy and forces the robot to stop. There are two kinds of stop states in our reactions: the touch reaction and the stop reaction. The stop reaction is basically an emergency stop command to prevent harm to a human or an environment. It also used a transition block to other reactions. The touch reaction activates only if the contact is with a human. This state is used to stop execution for a short period. The manual control can be implemented as reactions to touches: the first touch means the robot stop, the second one is a resume. It can be used to change the workpiece by a human, or correct the workpiece position. If contact is persisting and becomes continuous, the touch reaction becomes a stop reaction, and then for a soft (”S”) contact in the robot end-effector switches to the compliant mode. In the compliant mode, the position and orientation of the robot can be changed manually. The robot could be guided by

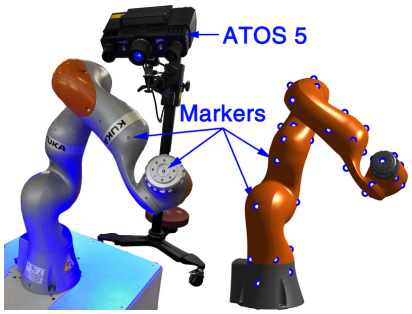


Fig. 5. Real and simulated KUKA iiwa with ATOS5 markers. By measuring markers coordinate is a robot base frame, it is possible to match markers position in the real robot and its virtual model, to obtain ground truth for localization.

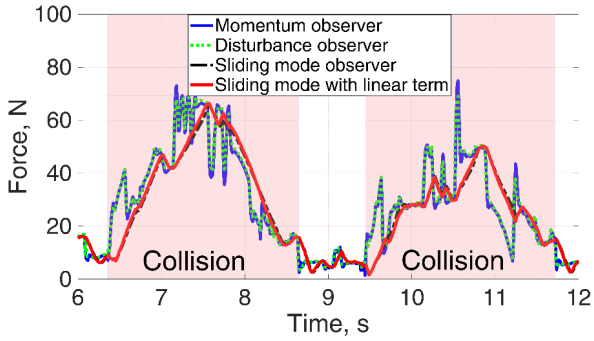


Fig. 6. Force estimation with different types of observers for UR10 robot during the contact with a human.

a human to the next screw position if it was unrecognized by the vision system. Also if there was accidental contact with a human, firstly robot will stop, then, if the collision still exists, become compliant. In this way, we try to limit interaction force for safety.

In the case where something touches the robot in the intermediate point or "elbow" ("EL"), the robot enables elbow reaction. In this mode, the robot will try to use its kinematic redundancy to avoid the collision, while following the desired trajectory by the end-effector. If the end-effector ("EE") itself has collided with a hard object ("H") during the non-contact operation, we also could try to avoid this obstacle. In our example, the robot could have a collision between its tool and a workpiece/environment during the inspection phase. To implement this, we used a potential field approach, where a point with a collision in the base frame was added as a new point with high potential.

The states and transitions in the finite state machine presented in this section could be extended or altered to better describe robot behavior during multi-scenario contact operations.

IV. Experiments and Results

A. Hardware and Datasets

For the experimental study, two collaborative robots were used: KUKA iiwa 14 and Universal Robots UR 10e. The obtained results were tested in simulation and hardware.

We used two separate datasets, one for classification and the other for localization. The dataset for classification consists of 430000 frames with a 100Hz rate, with a total of 300 random trajectories and 500 collision events. It includes the collision data with hard dynamic objects, hard static objects, and three different persons. The contacts with the humans occurred not only in a collaborative way, where a human guided the robot, but also in an accidental way, where the robot unintentionally collided with a human's arm, back, or body.

The second dataset for localization consists of 100 random robot configurations for 22 contact points, 2200 samples in total. For the real robot, only 10 test configurations were used to verify our results. All points of contact were located in the last four links where a random force from a friction cone was applied. Contact point location was estimated with the ATOS 5 measurement system with less than millimeter accuracy, all 22 contact points were marked on the real robot and its model in simulation as shown in Fig. 5. During the experiment on the real robot, the operator applied external force for the marked point with a known location on the robot surface. The ground truth for force vector amplitude or direction is not provided, but in the future, we plan to use a special tool with a force sensor.

The developed algorithm was tested on Intel Core i5-4210H 3GHz CPU, 8Gb RAM PC as a Matlab 2-thread program.

B. Contact Detection Results

We have tested three algorithms on the UR10e robot: momentum observer, disturbance observer, and sliding mode momentum observers. This robot is not equipped with torque sensors in each joint but allows to measure motor currents. After that, the current-to-torque matrix can be used to estimate the load in motors. Each observer has its parameters for configuration. We used the Fmincon optimization technique to find the optimal values. More information about dynamic parameters identification and observers can be found in [31].

During the experiment, we pushed the robot in different positions and measured angles, velocities, and currents. The result of external force estimation is shown in Fig. 6.

In general, all observers have demonstrated similar output. In terms of the detection speed, the results of the momentum and the disturbance observers are slightly better than the sliding mode. From a practical point of view, the disturbance observer algorithm is more compact for implementation but requires the inversion of the robot inertia matrix and configuration of three values regardless of the joint number. Momentum observer does not use matrix inversion but requires a transposed version of the matrix with centripetal and Coriolis terms. The amount of parameters for configuration is equal to the joint number. Thus the algorithms are almost equal and the choice could depend on secondary factors.

To estimate part of torque that was generated only by the external force for KUKA iiwa and UR10, we used

TABLE I

LOCALIZATION ALGORITHMS RESULTS FOR SIMULATED AND REAL ROBOT KUKA IIWA 14.

Noise for torque measurements	Simulated results 0		Simulated results $N(0, 0.5Nm)$		Real experiment *			
Method for collision localization	error $\Delta d, m$		error $\Delta d, m$		error $\Delta d, m$		Run-time, ms	
	Average	std.	Average	std.	Average	std.	Average	std.
Proposed Particle Filter on Graph (PFG)	0	0.01	0.04	0.06	0.04	0.05	17	3
Two-step Optimization (TS)	0	0.05	0.05	0.08	0.06	0.09	10	5
Feed-forward NN (NN)	0.05	0.09	0.08	0.13	0.09	0.08	5	2
Direct Optimization (DR)	0.05	0.11	0.08	0.11	0.07	0.09	52	6

moment observer. It provides slightly better time response characteristic compared to the two other methods.

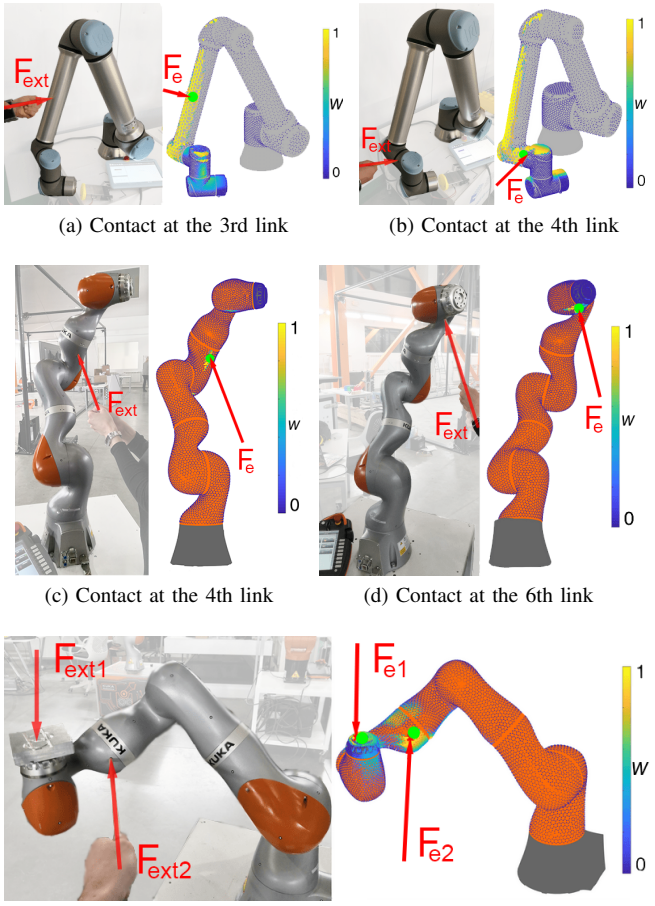
C. Contact Localization Results

Four localization algorithms were chosen to compare their performance. To make a fair comparison, the single contact situation is considered. The first algorithm is a proposed modification of particle filter on the graph (PFG) with 50 particles, the second one is a two-step optimization (TS) algorithm from [17], the third one is a one-layer feed-forward neural network with 100 neurons in a hidden layer (NN) and the fourth one is Direct-based optimization (DR) from [11]. The first two approaches use information about the robot surface and find a force inside a friction cone, the rest two approaches use a cylindrical approximation of a robot shape and assume an external force normal to the link.

The results of the simulations and tests on the real hardware are presented in Table I. Without noise in torque measurements, PF and TS algorithms gave almost zero error in position and external force. With added noise, these algorithms showed 4-5 cm mean error in position estimation and 16-17 % error in external force. With the real robot, PF gave slightly better localization than a TS but had a slower update loop, 60 Hz vs. 90 Hz. In practice, a larger mean error of TS can be explained by obtaining a local minimum in the wrong link.

Two other approaches, NN and DR showed more than 10 cm mean error even without noise. It could be explained by the complex shape of a robot surface, which these algorithms neglect. The force estimation has a large error due to the assumption of force acting on the normal. The machine learning-based approach is the fastest of the studied methods and can be used when the precise location of contact is not very important. NN can also be used for the classification of a collision point from a set of predefined locations.

The examples of PFG estimation for real KUKA iiwa and UR10e robots are presented in Fig. 7 (a-d). The multi-collision case is shown in Fig. 7 (e), here the first external force is caused by the constant gravity force applied to the end-effector, and the second force applied by a human.



(a) Contact at the 3rd link (b) Contact at the 4th link

(c) Contact at the 4th link (d) Contact at the 6th link

(e) Multi-collision case, with contact at EE and 5th link

Fig. 7. Localization algorithm for KUKA iiwa and UR10e. Probability at all points is estimated for visualization. The green dot is a collision location, the red arrow is the estimated contact location and force direction.

D. Contact Classification Results

The classification of hard/soft contacts is performed using deep neural networks. To train the network for collision type classification, the previously mentioned dataset was cut into samples with a window of 1 second with overlap. Each sample has a size of 100 by 14, where 100 is the window size, and 14 is the number of values with torques and positions for robot joints. After the partition, an undersampling technique was used to balance data by classes, resulting in 3300 samples of each class in the dataset. The neural network was trained using the backpropagation algorithm with Adam optimizer. Binary cross-entropy was used as a loss function. The trained neural network showed an accuracy of 98.8% on the test data compared to 89.5% from a previous frequency based approach [12]. In practice, the algorithm was able to

reliably separate contacts with a human's hand, arm, or body from contacts with hard objects, such as metal workpieces, walls, wooden furniture.

V. Conclusion

In this work, we presented a multi-scenario contacts handling methodology for collaborative robots. It includes and considers four main stages of collision event: detection, localization, classification, and reaction. For the detection stage, state-of-the-art observers were implemented and tested. The localization of multiple contacts was performed using a particle filter on the graph which utilizes a graph representation of a remeshed robot link to improve computational efficiency. The classification module includes six different categories of interaction where soft/hard contact classification was carried out using a deep neural network. For the reaction stage, a finite state machine with states as behavior scenarios and transitions as contact events was presented. All stages were tested in a simulation environment and with a real collaborative robot KUKA iiwa, and some stages with the UR10e robot.

In the experimental study, three different observers were implemented, and four localization algorithms were compared. The result of the proposed localization algorithm showed a 4 cm mean error for the last 4 links of the KUKA iiwa robot. The contact nature classification achieved 98% accuracy. All algorithms were capable of working in real-time with 50Hz frequency.

References

[1] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, "An atlas of physical human–robot interaction," *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253–270, 2008.

[2] T. Harden, K. A. Schroeder, and M. W. Pryor, "On the use of joint torque sensors for collision detection in a confined environment," tech. rep., Los Alamos National Lab., Los Alamos, NM, USA, 2011.

[3] S. Takakura, T. Murakami, and K. Ohnishi, "An approach to collision detection and recovery motion in industrial robot," in *15th Annual Conference of IEEE Industrial Electronics Society*, pp. 421–426, 1989.

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

[5] A. De Luca and R. Mattone, "Sensorless robot collision detection and hybrid force/motion control," in *Proceedings of IEEE international conference on robotics and automation*, pp. 999–1004, 2005.

[6] T. Tomić, C. Ott, and S. Haddadin, "External wrench estimation, collision detection, and reflex reaction for flying robots," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1467–1482, 2017.

[7] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292–1312, 2017.

[8] Y. J. Heo, D. Kim, W. Lee, H. Kim, J. Park, and W. K. Chung, "Collision detection for industrial collaborative robots: A deep learning approach," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 740–746, 2019.

[9] S. K. Ralph and D. K. Pai, "Detection and localization of unmodeled manipulator collisions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 2, pp. 504–509, 1995.

[10] G. Buondonno and A. De Luca, "Combining real and virtual sensors for measuring interaction forces and moments acting on a robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 794–800, 2016.

[11] N. Likar and L. Žlajpah, "External joint torque-based estimation of contact information," *International Journal of Advanced Robotic Systems*, vol. 11, no. 7, p. 107, 2014.

[12] D. Popov, A. Klimchik, and N. Mavridis, "Collision detection, localization & classification for industrial robots with joint torque sensors," in *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 838–843, 2017.

[13] A. Zwiener, C. Geckeler, and A. Zell, "Contact point localization for articulated manipulators with proprioceptive sensors and machine learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 323–329, 2018.

[14] L. Manuelli and R. Tedrake, "Localizing external contact using proprioceptive sensors: The contact particle filter," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5062–5069, 2016.

[15] J. Bimbo, C. Pacchierotti, N. Tsagarakis, and D. Prattichizzo, "Collision detection and isolation on a robot using joint torque sensing," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[16] A. Zwiener, R. Hanten, C. Schulz, and A. Zell, "ARMCL - ARM Contact point Localization via Monte Carlo Localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2019.

[17] D. Popov and A. Klimchik, "Real-time external contact force estimation and localization for collaborative robot," in *IEEE International Conference on Mechatronics (ICM)*, vol. 1, pp. 646–651, 2019.

[18] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp, "Reaching in clutter with whole-arm tactile sensing," *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 458–482, 2013.

[19] F. J. A. Chavez, J. Kangro, S. Traversaro, F. Nori, and D. Pucci, "Contact force and joint torque estimation using skin," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3900–3907, 2018.

[20] E. Magrini, F. Flacco, and A. De Luca, "Estimation of contact forces using a virtual force sensor," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2126–2133, 2014.

[21] E. Magrini, F. Ferraguti, A. J. Ronga, F. Pini, A. De Luca, and F. Leali, "Human-robot coexistence and interaction in open industrial cells," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101846, 2020.

[22] A. De Luca and F. Flacco, "Integrated control for phri: Collision avoidance, detection, reaction and collaboration," in *IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pp. 288–295, 2012.

[23] S. Golz, C. Osendorfer, and S. Haddadin, "Using tactile sensation for learning contact knowledge: Discriminate collision from physical interaction," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3788–3794, 2015.

[24] A. Kouris, F. Dimeas, and N. Aspragathos, "A frequency domain approach for contact type distinction in human–robot collaboration," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 720–727, 2018.

[25] S. Parusel, S. Haddadin, and A. Albu-Schäffer, "Modular state-based behavior control for safe human-robot interaction: A lightweight control architecture for a lightweight robot," in *IEEE International Conference on Robotics and Automation*, pp. 4298–4305, 2011.

[26] E. Magrini and A. De Luca, "Human-robot coexistence and contact handling with redundant robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4611–4617, 2017.

[27] S. Mikhel, D. Popov, and A. Klimchik, "Collision driven multi scenario approach for human collaboration with industrial robot," in *Proceedings of the International Conference on Mechatronics and Robotics Engineering*, pp. 78–84, 2018.

[28] S. Mikhel, D. Popov, S. Mamedov, and A. Klimchik, "Development of typical collision reactions in combination with algorithms for external impacts identification," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 253–258, 2019.

[29] A. Mohammadi, M. Tavakoli, H. J. Marquez, and F. Hashemzadeh, "Nonlinear disturbance observer design for robotic manipulators," *Control Engineering Practice*, vol. 21, no. 3, pp. 253–267, 2013.

[30] G. Garofalo, N. Mansfeld, J. Jankowski, and C. Ott, "Sliding mode momentum observers for estimation of external torques and joint acceleration," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6117–6123, 2019.

[31] S. Mamedov and S. Mikhel, "Practical aspects of model-based collision detection," *Frontiers in Robotics and AI*, vol. 7, p. 162, 2020.