Touch driven controller and tactile features for physical interactions

Zhanat Kappassov, Juan Antonio Corrales Ramon, Véronique Perdereau

Touch driven controller and tactile features for physical interactions

Zhanat Kappassov, Juan Antonio Corrales Ramon, Véronique Perdereau

Touch driven controller and tactile features for physical interactions

Zhanat Kappassov[a,*], Juan-Antonio Corrales[b], Véronique Perdereau[c]

[a] *Department of Robotics, Nazarbayev University, Kabanbay batyr ave. 53, 010000, Astana, Kazakhstan*
[b] *Sigma-Clermont, CNRS, Institute Pascal, F-63000 Clermont-Ferrand, France*
[c] *Sorbonne Universités, UPMC Univ Paris 06, CNRS, ISIR, 4 Place Jussieu, 75252, Paris, France*

Abstract

We propose an approach that considers controlling contact between a robot and the environment during physical interactions. Current physical interaction control approaches are limited in terms of the range of tasks that can be performed. To allow robots to perform more tasks, we derive tactile features representing deformations of the mechanically compliant sensing surface of a tactile sensor and incorporate these features to a robot controller, akin to a visual servo, via touch- and task-dependent tactile feature mapping matrices. As a first contribution, we derive tactile features to localize a contact coordinate frame between an object and an array of pressure sensing elements, with a mechanically compliant surface, attached onto a robot arm end-effector interacting with the object. As a second contribution, we propose tactile projection matrices to design a tactile servoing controller that combines these tactile features with a Cartesian impedance controller of the robot arm. These matrices convert the proposed tactile features to balance not only normal forces but also torques about the sensor's axes. It allows the end-effector to steer the contact frame in a desired manner by regulating errors in the tactile features to address several common issues in robotics: exploration and co-manipulation.

Keywords: Robot Arm Control, Physical Interaction, Tactile Sensing Arrays, Tactile Servoing, Manipulation, Haptic Exploration

1. Introduction

Physical interaction tasks, including dexterous manipulation, object exploration and recognition, can be performed even when a manipulated object is occluded from vision sensors. Indeed, the dexterity of manipulation is not affected by the blindness of a person ([1]). But, in the absence of the sense of touch (tactile feedback), contact forces generated during the interaction with the environment can only be estimated roughly. Then, the performance of a manipulation task depends on hard-coded internal models of the environment. However, the real world is highly variant; it can not be assumed, therefore, that the environment is perfectly known. Nowadays robots move from fenced industrial spaces into this real human-inhabited environment [2], in which the physical interactions are essential. Exploration and manipulation require to handle the inherently occurring contacts between a robot and objects. Therefore, new algorithms based on signals from artificial tactile sensors (i.e. robotic sense of touch) are required in order to control these physical contacts of the robot body with the objects of the environment.

With progress in sensing technology, as reviewed in [3], tactile pressure sensing arrays have emerged [4]. This type of sensor can provide a pressure profile of a contact region thanks to their ability to detect multiple interacting points within one continuous region at a single time instance. This ability offers new research directions for allowing robots to interact with the environment more dexterously.

Most research on robot physical interaction using a tactile sensing array has focused on the execution of a single task [5, 6] or multiple disconnected tasks [7], whereas much less effort has been put into switching from one task to another within one application. During such transitions, properties of a contact between a sensor and the environment may change, e.g. point-contact to edge-contact. These two types of contact are dominant in object manipulation tasks [8] and, therefore, only edge and point contacts are considered in this work.

As a first contribution of this paper, we propose projection matrices, which depend on a task and on properties of a contact, to construct a tactile-position controller. This controller regulates the contact properties of a robot arm, rendering it more dexterous and flexible to interact with an unknown environment autonomously. As a second contribution, we propose a new tactile feature, that is derived using the center of pressure and center of contact, representing deformations of the contact surface. Indeed, in

*Corresponding author
Email addresses: zhkappassov@nu.edu.kz (Zhanat Kappassov), juan.corrales@sigma-clermont.fr (Juan-Antonio Corrales), veronique.perdereau@sorbonne-universite.fr (Véronique Perdereau)

most robotic applications where tactile signals from planar sensing arrays are used, the geometric deformations of the sensor's surface are neglected, e.g. as in [7]. Our idea is to define them with rotational and translational compliance variables for incorporating them into the controller.

We show with experiments that this enhances the robots performance in maintaining contact with unknown and unpredictable surfaces by using a tactile sensor. Thereby, it is possible to handle complex touch-driven exploration or manipulation tasks under uncertainty as the tactile feedback can guide the robot to conform with the sensed shape of an object. The developed controller structure can be embedded with industrial robotic manipulators and service robots (Fig. 1).

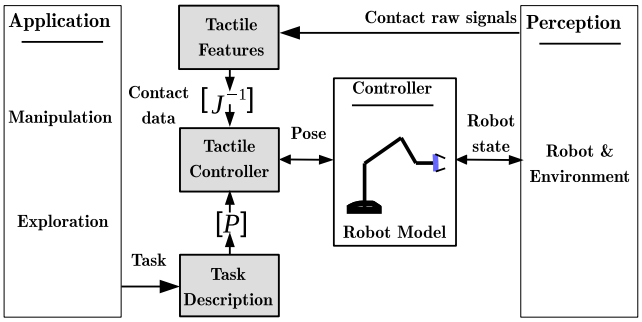

Figure 1: Overview of the methodology developed for achieving control of contact frame and transitions between tasks.

2. Related work

In this section, we will briefly review the literature on tactile servoing. Compared to the state-of-the-art approaches, our new contributions are: 1) the computation of $DZMP$ features to balance torques about the sensor's x and y axes, 2) introducing new matrix elements to the state-of-the-art inverse tactile Jacobian and projection matrix. The first contribution is drawn on the ideas given in [9] and extends them for planar tactile arrays with an elastic surface. Especially, there were no tactile features to estimate a non-uniform line force applied to the sensing surface. The second contribution is drawn on the ideas of the tactile servoing framework [5, 7]. Our inverse tactile Jacobian and projection matrix incorporate new touch-dependent and task-dependent components. These components increase the number of controllable DOFs of the robot end-effector in edge servoing tasks. More precisely, an applied moment of force on an edge of an object has become controllable: 1) for not only the ideal case when the edge is fully covering the sensor but also for a partial contact when the edge in not covering the sensor from one side to the opposite, 2) so that the deviations of the force moment applied by the edge can act as the disturbance signals for both translational motion (e.g. co-manipulation task as described in Section 5.3), and rotational motion (e.g. aligning with an edge as described in Section 5.2).

2.1. Tactile feature extraction

In regard to robot applications different signal processing techniques have been applied to tactile signals in order to extract *tactile features* [10, 11]. Whereas some approaches are suitable for event detection, e.g. [12], others are focused on extracting features representing geometrical deformations of a contact surface of a planar tactile array, e.g. [8]. The tactile array can provide measures of such an important object property as an *edge* at a single snapshot. Though researchers in human tactile perception, e.g. [13], argue that an edge is detected with shear strain of the human skin, most of state-of-the-art commercial sensing arrays can only measure normal forces. In fact, there are only a few sensor prototypes based on optical technologies that are able to measure shear forces [14]. Fortunately, the distribution of these normal forces provide with information about two typical types of contacts that appear when common planar tactile sensors touch an object: point (Fig. 2) and edge (Fig. 3) contact on a plane. Since the time when the spatial resolution of planar pressure sensing arrays allowed to detect these types of contact [15], studies on the extraction of tactile features representing the deformations of a contact surface have gained attention thanks to robot applications involving exploration and manipulation. The first approaches were heading the detection of an edge in a tactile image [16]. The computational costs were important, especially, when the processed signals were used in real-time applications. Thus, in [17] a faster algorithm was proposed. The algorithm included automatic threshold operation, edge detection, and Hough Transformation to fit a line in the sensor image that yield the orientation of this line with respect to the sensor's coordinate frame. Since the tactile image was thresholded, a total force was represented by the area of the sensor covered by the object. The relationship between the estimated and real forces was not discussed. With emphasis on the drawbacks of the thresholding operation, which could lead to large errors when the threshold is wrong, Chen and Zhang [18, 5] proposed a more approach based on the analysis of the subsurface strain and stress. Under a number of assumptions, the authors derived four tactile features: the normal force, the two coordinates of the contact location, and the contact orientation. The edge was assumed to be infinitely long and to have intersects with two opposite edges of the tactile sensor. Moreover, computations of strain and stress distributions were still complex and a number of physical constraints had to be obeyed. In order to mitigate the computational complexity, [9] derived quantitative information about the location and orientation of a contact (for the edge-contact type) by exploiting image moments. Thus far, only four tactile features have been used to describe the location and orientation of a contact in the case of an edge contact. However, the edge should be localized with six features since a line - force applied by the edge may have a non-uniform distribution around the planar sensing surface and, therefore, the edge will cause rotational deformations of the surface

(Fig. 3(c)). For quantitative estimation of such deformations caused by the non-uniform line-force, a novel tactile feature is needed. The mentioned works did not consider this type of contact due to the following reasons:

- In [16] and [17] thresholding is used before applying feature extraction algorithms, so the deformations of the sensing surface due to an external moment of force applied by an edge can not be estimated.
- In [5] due to the complexity of calculations, it was assumed that the applied forces along an edge were uniformly distributed.
- In [9] the tactile pressure sensing array was rigid and did not incorporate an elastic surface.

In order to overcome this drawback, we propose an algorithm, extracting a new distance-to-zero-moment-point ($DZMP$) tactile feature, that allows a robot to control rotations of its end-effector during physical interactions. The algorithm is based on the calculation of the distance between the center of pressure and the center of contact. This distance represents a moment of force, which is proportional to rotational displacements and the rotational compliance constants of a tactile array. This new tactile feature will be described in detail in section 3.2.

2.2. Tactile servoing

The concept of tactile servoing is analogous to image-based visual servoing [19]. The difference is in the way of interaction with the environment – a robot is in contact with it. The thrust, that pulled the control community towards the incorporation of tactile feedback, was due to the fact that image cameras have a particular drawback: they produce a two-dimensional representation of a three-dimensional scene whereas tactile pressure sensing arrays can produce a fine one-to-one representation of the part of the scene at a single time instant [20]. As in visual servoing [19], it is then possible to reconstruct the shape of an unknown object by sliding over it in both discontinuous and continuous ways using non-planar and planar tactile sensing arrays.

2.2.1. Touch driven control using non-planar sensors

Sliding over unknown surfaces is an essential task in object exploration. In force control literature, it was implemented using non-planar sensors. For example, in [21], the authors tackled the problem using a robot with an ellipsoidal shaped force sensor driven by a force control algorithm. As a result they reconstructed the contour of the explored object's surface. The contact location with the environment was estimated by measuring forces and torques. Such approaches require exact friction properties of the surfaces in contact. Moreover, when there is a need to explore the edge of an object, the force sensors can not be applied because they are limited to a single contact. When attached to several links of a robot hand, this drawback of single contact measuring sensors is mitigated [22]. Multiple contacts can be measured with tactile sensing arrays. In [23] and [24], a tactile sensing array covering a robot fingertip and a hemispherical optical tactile sensor, respectively, were used to follow an edge of an object by performing palpations. Nevertheless, palpations unavoidably lead to the undesirable discontinuous contacts.

2.2.2. Servoing with planar sensors

As the touch driven control algorithms are important in exploratory actions, tactile servoing plays an essential role in exploration of unknown objects. Tactile servoing allows continual contact with the environment rather than palpations. However, due to imperfections of tactile sensing arrays and difficulties in integration of these arrays, tactile servoing has been implemented on planar sensors only. In [25], the author illustrated the efficacy of using a tactile sensor – the sensor was developed in 1980s and described in [4] – for detecting contact and for complementing a visual system to distinguish whether circumferences detected by a 2D vision belong to holes or concavities. However, the motion of the robot was not driven by the tactile feedback. One of the first robot motion driven by using a tactile sensing array was implemented by [17] in the edge tracking problem. The sensor array provided a tactile image that was considered as a grayscale vision image. The orientation of the edge with respect to the sensor coordinate system was obtained using the Hough Transform. This edge was tracked by a hybrid force and position controller [26]. [27] developed a tactile based control structure that allowed a robot to not only slide over an object edge, as in the latter approach, but also to manipulate objects by rolling them. In contrast to the previous approach, the authors applied morphology analysis in order to derive zero-, first-, and second- order moments of the contact pressure distribution. This increased the number of controllable degrees of freedom (DOF) of a robot end-effector. However, only the edge contact has been considered so far. Drawing on these ideas, Chen and Zhang [18, 5] extended the concept to a more general approach and performed tactile servoing by introducing an inverse Tactile Jacobian for edge- and point- contact types in order to map errors in tactile features to an error of the robot end-effector. Contact location and contact angle were extracted from a tactile image to perform an edge following task with a number of assumptions, including that the edge is infinite. The limitation of the approach was that there was no flexibility of the controller to switch between different contact types when a robot interacts with a point or edge of an object. Li et. al [7] came up with a unified tactile servoing framework that allowed switchable control of different servoing modes for the edge and point contact types. In contrast to the previous approach, the authors applied the Principal Component Analysis (PCA) to extract the orientation of the edge with respect to a sensor coordinate system. Four tactile features were used in order to artificially constrain

the corresponding motions. The authors modified the inverse Tactile Jacobian matrix so that it mapped errors in four tactile features into errors in six components of the motion twist in the 6D Cartesian space (with 3 rotational components). The switch between different tasks was realized via a selection matrix. Nevertheless, on a planar sensing surface, the point-contact may transform to the edge-contact type (and vice-versa) during one task execution. Such transformations were not considered. Reactive response to them could be required in manipulation tasks as:

1 - when the edge is straight but not parallel to the sensing surface, so that, when a robot is in contact with the edge, there is only part of the edge in the tactile image. The robot would sense the point contact type instead of the edge. Therefore, the robot end-effector should first align with the edge in order to increase the contact area and finally find the edge-contact. For this case, the robot should be able to detect the edge when it appears in the tactile image and consequently switch to the next task, e. g. the edge following task. In the existing tactile servoing approaches, this was not performed yet. This case is illustrated in Fig. 4.

2 - when the edge is not infinite in the tactile image. Imagine that, in the previous case, when the end-effector was aligning with the edge, the edge was short so that only a "*partial*" edge appears in the tactile image. By "*partial*" edge, we refer to the edge when it is not covering all of the sensing surface, i.e. when a line in the tactile image is not infinite. Then, during the alignment with the edge, the previous approaches would fail. This case is illustrated in Fig. 3(a).

3 - when the edge is bent in the 3D Cartesian space and not only in one plane. In the baseline approaches, this case was not considered because the moment of force applied to the edge could not be estimated (Fig. 3(c)).

In summary, the main drawback of the baseline approaches is that they are limited to specific contact configurations between objects and tactile sensors (Table 1). Section 5.2 describes several experiments of the listed contact tasks that were not possible before. Thus, the functionality and dexterity of manipulators were limited. The number of DOFs that can be controlled based on the tactile feedback in the baseline approaches was limited to 4 instead of 6 DOF of our approach for the case of the edge-contact type. In order to overcome the disadvantages of the baseline approaches, a new **touch-dependent inverse tactile Jacobian** and **task projection matrix** are proposed in the paper. The tactile Jacobian incorporates new tactile features and allows a robot to perform complex manipulation tasks when contact types change during the execution.

Table 1: Summary of the state-of-the-art in the tactile servoing.

Controller	N DOF edge	N DOF point	Transition between contact types.
[17]	2	(×)	(×)
[27]	3	(×)	(×)
[18, 5]	3	3	(×)
[7]	4	3	(×) Selection matrix without active detection of a contact type
Our	6 DOF	3 DOF	Detection of edge and point

3. Tactile feature extraction

A pressure sensing array produces a two-dimensional pressure profile. Such an array (Fig. 2(a, b)) is constructed by load sensing elements (*tactels* or *sensing cells*). Each element measures the pressure caused by an object pressed against the tactel. And the whole array of the tactels provides a spatial distribution of the pressures. This pressure profile preserves the local contact information such as the magnitude of an applied normal force, and the contact location and orientation with respect to the pressure sensor. An online tactile image processing is necessary to extract this local contact information for moving a robot with a tactile servoing controller (Sec. 4). More precisely, the information is represented by a vector of tactile features. This vector defines the Cartesian location of the origin of a *contact coordinate frame* so that its base axes coincide with the sensor frame. The number of entries in the tactile feature vector defining the contact coordinate frame is different for point and edge types of contact. Each contact type is described in the following of this section.

3.1. Point contact

There are many cases when the contact between the sensing surface and an object can have the point-contact type, e.g. a stiff spherical object applying a normal force onto a sensing array with an elastic layer. The origin of the sensing frame is located at the geometric center of the sensing surface, and it lies on the surface of the elastic layer as shown in Fig. 2(a). Two components of the location of the contact coordinate frame $\{c\}$ are captured by the pressure array: x_c and y_c are the x- and y-components of the location with respect to sensor coordinate frame $\{s\}$. The sensing layer has the compliance constant C_z along z axis. Then the deflection d_z of the contact frame along the axis is given by $d_z = C_z f$, where f is the total force applied by the spherical object. Every tactel that is under the object measures the applied force. It is then reasonable to plot the sensor measurements as a gray scale image where the intensity of a pixel corresponding to an individual tactel is proportional to the force at that location (Fig. 2(d)). From the resulting gray scale image, we can extract the tactile features representing the contact frame by exploiting image moment analysis Eq. (B.1). The first two tactile features are the center of pressure (CoP) along x-axis (CoP_x) and y-axis (CoP_y), that are given by Eq. (B.2) and (B.3), respectively. Fig. 2(d) illustrates this center (red cross) overlaying the tactile image of the contact. In the Cartesian space, these two tactile features correspond to two

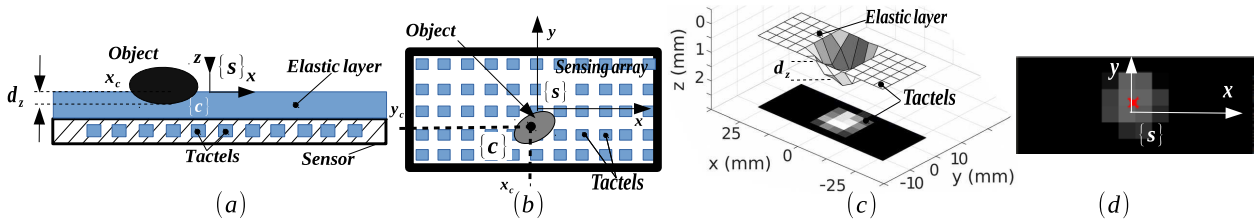

Figure 2: A point contact with a pressure array with elastic surface. An object under external force deforms the sensing surface, which is shown from (a) the side view and (b) the top view. (c) The corresponding deformation of the sensor surface (up) with the pressure profile (bottom). (d) Gray scale representation of the pressure distribution. Coordinates x_c, y_c and d_z of the contact frame $\{c\}$ represent the center of the contact area with respect to the tactile sensing frame $\{s\}$.

coordinates (x_c, y_c) of the contact coordinate frame shown in Fig. 2(b). Even though we compute the coordinates of the center of pressure, we can assume that it coincides with the center of contact because the highest pressure is at the middle of the contact area for point-contact. This is not the case for edge contact discussed in the next section. The third coordinate $z_c = d_z$ is proportional to the total normal force and shows how deep the contact frame is deformed from a resting state (Fig. 2(a, c)). This force can be estimated by the zeroth moment $f_z \propto M_{0,0}$. Since the sensing layer is elastic, the area of contact A increases with force. In this connection, we can also assume $f \propto A$. Thus far, the location of origin of the contact frame for the point-contact type is given by CoP and d_z:

$$\boldsymbol{O_c} \propto [CoP_x, CoP_y, d_z]^T \tag{1}$$

where $\boldsymbol{O_c}$ is the origin of the contact frame.

3.2. Edge contact with uniform and non-uniform load

The edge contact type can occur during blind exploration of objects with rectangular and cylindrical shapes. And the elastic sensing surface can comply with an edge so that it can be under a uniform or non-uniform force distribution along the edge. Fig. 3(a) depicts a stiff cylindrical object pushing a sensing array with a compliant surface. When the stiff cylinder applies a normal force on the sensing surface, there are multiple tactels in contact with it. They create a contact region, whose center represents the origin of the contact frame $\{c\}$ given by the three components, defined in Eq. (1), and the orientation of the edge about z-axis, given by an angle α_z between the main principle component and the x-axis of the sensing frame $\{s\}$.

The orientation of the edge is given by the principle component, obtained from the PCA technique, that represents the direction in which the contact points are spread the most (Fig. 3(e)). This principal component U_1 is the eigenvector having the highest eigenvalue λ_1. Then the orientation of the edge is given by the angle α_z, derived in Eq. (B.5), between this principal component and the basis vector defining the x component of the sensing frame.

So far, it was assumed that the pressure sensing array was in contact with a cylindrical object that was under an external force with a uniform distribution along the edge (Fig. 3(a)). Let us consider now the case when the line force applied onto the edge is non-uniform as shown in Fig. 3(c). Then, in addition to the previously defined components of the contact origin, including x_c, y_c, d_z, and α_z, there is one more geometrical transformation. This transformation is given by the Euler angle of rotation around x-axis d_{wx} or y-axis d_{wy} that describes a non-uniform line load.

Being intuitive and simple, Euler angles representation of rotations has one drawback – known as gimbal lock that happens when rotation around one of the three orthogonal axes coincide with another axis, for instance when $d_{wx} = 90$ degrees. Specifically, the derivative of Euler angles representation $R^3 \rightarrow SO(3)$ does not have rank three in the exemplified case, which leads to unsolvable inverse transformation that is needed at the controller level (Section 4). This drawback is mitigated as the deformations of the sensing surface never reach 90 degrees and α_z can be measured in modulo π in the range $(-\pi/2, \pi/2)$ to circumvent singularities.

The **non-uniform line load** should be recovered from

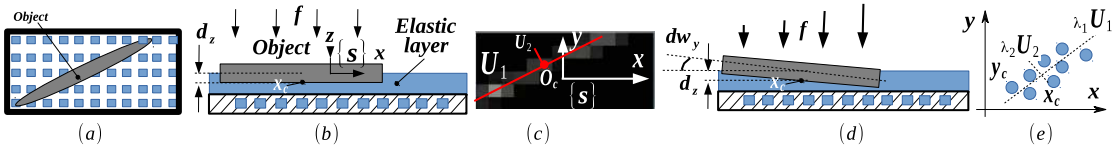

Figure 3: An edge contact with a pressure array with elastic surface. An object (a) under a uniformly (b) and (d) non-uniformly distributed external force deforming the sensing surface. Coordinates x_c, y_c and d_z of the contact frame $\{c\}$ represent the center of the area of contact with respect to the tactile sensing frame defined as $\{s\}$. The non-uniform line load force deforms the sensing surface so that it causes inclinations of the contact frame. (c) Gray scale representation of the pressure distribution with the major principal component axis. (e) Principle Component Analysis. Two orthogonal eigenvectors U_1 and U_2 and their corresponding eigen values λ_1 and λ_2 with $\lambda_1 > \lambda_2$ are obtained.

the flat pressure sensing array, measuring only normal pressures, so that there will be six features defining the contact coordinate frame in the 6D Cartesian space. This suggests taking into account rotational compliance constants C_{w_x} and C_{w_y} about x- and y-axes, respectively. Therefore, we can model the elastic surface as a general 3D spring with compliance vector $\boldsymbol{C} = [0, 0, C_z, C_{w_x}, C_{w_y}, 0]^T$ that includes one translational (C_z) and two rotational (C_{wx} and C_{wy}) compliance constants.

In case of the edge with non-uniform line load, the contact coordinate frame $\boldsymbol{O_c}$ is the frame that is reoriented with respect to the sensing frame $\{s\}$ due to not only the normal force f_z but also the moments of force μ_x or μ_y. Assuming that friction forces are relatively small, the deformations along z-axis and about x- and y-axes are defined by these compliance constants and construct the contact coordinate frame:

$$\begin{aligned} \boldsymbol{O_c} = [0, 0, dz_c, dw_x, dw_y, 0]^T = \\ [0, 0, C_z, C_{w_x}, C_{w_y}, 0]^T \cdot [0, 0, f_z, \mu_x, \mu_y, 0] \end{aligned} \tag{2}$$

DZMP tactile features proposed in this work represent these moments of force. First, the CoP_x and CoP_y tactile features are derived according to Eq. (B.2). Then the original tactile image is binarized with the lowest possible threshold. The threshold value can be experimentally tuned. It is also possible to automatically select the threshold value as it was shown in [28]. After binarization, the tactile image represents the area of contact: any tactel that is in contact with the environment gets the maximum gray-scale value. In this binarized image, the first order moments $M_{1,0}$ and $M_{0,1}$ are calculated according to Eq. (B.2). Since the tactile image was already binarized, the results of the equation represent the center of contact (CoC_x along x-axis and CoC_y along y-axis), not the center of pressure of the contact pattern. Before the image was binarized, the resulting image moment is weighted towards areas with higher pressure values. Assuming that rotational compliance constants in $\boldsymbol{C}$ do not vary, the Euclidean distance between these two different centers (CoP and CoC) is proportional to the rotational displacement of the contact frame due to the applied moment of force:

$$\begin{aligned} dw_x \propto DZMP_x = CoP_y - CoC_y \\ dw_y \propto DZMP_y = CoP_x - CoC_x \end{aligned} \tag{3}$$

where dw_x and dw_y are the rotational displacements from the resting state around x- and y- axes, respectively.

Finally, the six features represent each Cartesian coordinate of the contact frame for the edge-contact type:

$$\begin{aligned} \boldsymbol{O_c} = [x, y, z, \omega x, \omega y, \omega z]^T \sim \\ [CoP_x, CoP_y, d_z, DZMP_x, DZMP_y, \alpha_z]^T \end{aligned} \tag{4}$$

3.3. Edge- and point- contact type identification

Recall that in Eq. (B.5) the PCA was applied to a tactile image to detect two eigen vectors U_1 and U_2. Then the resulting principle components can be used to differentiate the two types of contact. The inequality:

$$\lambda_1 >> \lambda_2 \tag{5}$$

being satisfied (Fig. 4(right-hand side)) means that there is one component at least multiple times larger than the other one. Thereby, the contact pattern can be considered as an edge and the corresponding eigen vector will determine its direction. Otherwise, there is a point-contact type in the tactile image despite the size of the contact area (Fig. 4 left-hand side). λ_1 and λ_2 are the eigenvalues corresponding to U_1 and U_2, respectively.

Figure 4: A point-contact type and an edge-contact type on the left-hand and right-hand sides, respectively. The ratio between the lengths of first two principal components in the tactile images are different for these two types of contact.

The previously derived tactile features reduce the dimension of the vector space to control. This space includes the following features:

$$\boldsymbol{s} = [CoP_x, CoP_y, d_z, CoC_x, CoC_y, \alpha_z]^T. \tag{6}$$

4. Robot arm tactile controller

There are two situations of physical interaction [5]: a robot is in contact with the fixed environment (Fig. 5(a)) and it is interacting with a movable object (Fig. 5(b)). This section describes the tactile servoing controller and the proposed in this work inverse tactile Jacobian and projection matrix that can be implemented in a robot arm for these two situations. The described above tactile feature vector is fed into the controller using the projection matrix. This matrix is needed for switching between two different tasks, e.g. rotating around an object vertex and sliding over the object. Given a desired and current feature vectors – the number of entries in the vector is different for point- and edge-contact – the Jacobian maps an error between them into controller set points. Therefore, the tactile feature vector $\boldsymbol{s}$ is transformed to either the tactile feature vector $\boldsymbol{s_{point}}$ representing the contact frame of the point-contact Eq. (1) or $\boldsymbol{s_{edge}}$ representing the contact frame of the edge-contact Eq. (4). Later, the errors of these tactile feature vectors are mapped by the projection matrix into translational and rotational errors in the sensor frame $\{s\}$. Since the sensor is rigidly attached to the robot, these errors will be transformed into its end-effector frame $\{e\}$. These errors will be transformed into the base of the robot $\{g\}$ for moving it accordingly with a classical Cartesian impedance controller. All these steps are going to be described in detail in the next subsections. Thus,

the contact feedback would then guide the robot to move in a desired manner based on continuous signals from a tactile sensing array.

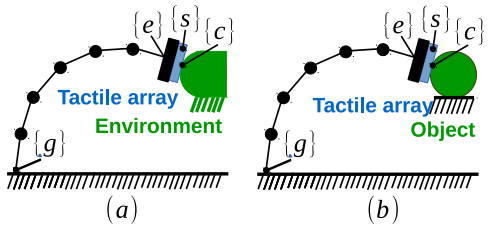

Figure 5: Physical interactions. A robot with a pressure sensing array interact with: (a) the stiff environment and (b) an object on the stiff environment. The frames $\{g, e, s, c\}$ correspond to the global, end-effector, sensor, and contact coordinate frames.

4.1. Feature error

The next step is to derive the tactile feature error. Applying an inverse tactile Jacobian and task projection matrix (Fig. 6) to the error will yield to a corrective Cartesian twist for the sensing coordinate frame. The error $\Delta \boldsymbol{s}$ is obtained by subtracting the current feature vector $\boldsymbol{s}$ from the desired feature vector $\boldsymbol{s_d}$.

$\boldsymbol{s_d}$ represents the desired contact coordinate frame and it depends on both the task to perform and the contact type. For example, in case of edge servoing task along x-axis $\boldsymbol{s_d} = [CoP_x^{des}, CoP_y^{des}, d_z^{des}, DZMP_x^{des}, -.\alpha_z^{des}]$, where superscript *des* indicates the desired signal. This result in the constrained motion with the sliding velocity υ_x along horizontal axis and the amount of applied force f_z so that the end-effector motion twist gets the following form $[\upsilon_x, 0, f_z, -, 0, 0]$.

$\Delta \boldsymbol{s}$ is given by:

$$\Delta \boldsymbol{s}_{edge,point} = \boldsymbol{s}_d - [\mathbf{J^{-1}}]\boldsymbol{s} \tag{7}$$

where $\mathbf{J^{-1}}$ is the inverse tactile Jacobian that can include either the $DZMP$ feature for an edge-contact type or the CoP feature for a point-contact type. This touch dependent inverse Tactile Jacobian is a matrix with following form and elements:

$$\mathbf{J^{-1}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -\beta & 0 \\ 1 & 0 & 0 & -\beta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \beta \end{bmatrix} \tag{8}$$

The variables $\beta \in \{0, 1\}$: $\beta = 1$ for the edge- contact type (for controlling the $DZMP_x$ or $DZMP_y$ features) and $\beta = 0$ for the point-contact type (this will allow rotation around a point contact (sections 5.2)). Shaded areas in Fig. 7 exemplify the purpose of the elements of $\mathbf{J^{-1}}$ and the contact state. For instance, first three diagonal components of $\mathbf{J^{-1}}$ are present in controlling the location and penetration depth of the point-contact. Control of the edge orientation, when line-force along the edge is uniform,

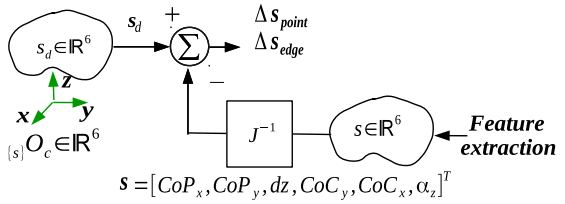

Figure 6: Feature error and touch-dependent tactile Jacobian.

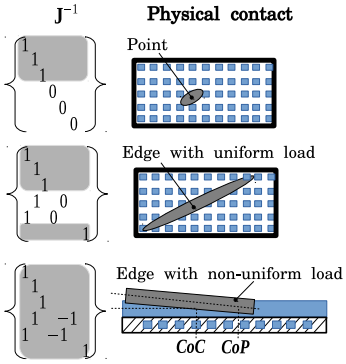

Figure 7: Illustration of elements of the touch-dependent tactile Jacobian for three physical contacts: point, edge, and edge with non-uniform load. The shaded rows in $\mathbf{J}^{-1}$ of every contact type contribute to the robot end-effector pose.

is realized with the first three and the last diagonal elements of the inverse Jacobian (second from top in Fig. 7). The fourth and fifth rows of this Jacobian are not involved in this case. The two entries in these rows contribute to the end-effector rotational motions (around x and y axes) performed for increasing the area of contact when an edge is expected to appear in the tactile image. Specifically, if the sensing surface is not parallel (i.e., there is a difference in the orientation with respect to x or y axis) to the edge of an object at the instant of a contact, a corrective motion is performed to eliminate the difference (see the description of Phase II in Section 5.2). Finally, when the edge is under non-uniform line force all of the elements contribute to the error fed into the next node in the tactile servoing controller.

The obtained error is multiplied with the projection matrix $\mathbf{P} \in \Re^{6\times6}$. Such operation allows the tactile servoing to switch between different tasks. Specifically, the error in CoP_x can be transformed to the translational or rotational error in the motion twist of a robot end-effector. The $\mathbf{P}$ matrix selects and maps feature errors so that the initial tactile feature error $\Delta\mathbf{s_{edge,point}}$ is mapped to the task-related feature error $\Delta\mathbf{s_{task}}$: |

$$\Delta \boldsymbol{s}_{task} = \mathbf{P} \cdot \Delta \boldsymbol{s}_{edge,point} \tag{9}$$

And $\mathbf{P}$ is given in the following equation.

$$\mathbf{P} = \begin{bmatrix} \gamma_x & 0 & 0 & 0 & \gamma_{com} & 0 \\ 0 & \gamma_y & 0 & 0 & 0 & 0 \\ 0 & 0 & \gamma_z & 0 & 0 & 0 \\ 0 & 0 & 0 & \gamma_{wx} & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma_{wy} & 0 \\ 0 & 0 & 0 & 0 & 0 & \gamma_{wz} \end{bmatrix} \tag{10}$$

where $\gamma_i \in 1, 0$ is a binary value that selects a set of features depending on the task. There is one special element γ_{com} that is also a binary value: when $\gamma_{com} = 1$ the rotational component around y axis of the tactile feature error is related to the translation error along x axis of the end-effector pose. This will allow to perform co-manipulation tasks as it will be shown in Section 5.3. Hence, the variables allow the use of the same controller in different tasks that have different kinematic constraints and even dynamic changes of the contact types.

Thus, equations Eq. (8) and (9) relate the error in the feature space to an increment of an end-effector pose in the Cartesian space.

4.2. External hybrid tactile-position controller

In the robotics literature, the "proportional-derivative (PD) plus gravity compensation" motion control law has been shown to be sufficient to perform a Cartesian control of a robot arm by applying $n \times 1$ joint torques [29] $\boldsymbol{\tau_d}$:

$$\boldsymbol{\tau_d} = \mathbf{J}_\theta^{\mathbf{T}}(\boldsymbol{K_{Pos}}(\boldsymbol{x_d} - \boldsymbol{x}) - \boldsymbol{K_D}\dot{\boldsymbol{x}}) + \boldsymbol{g(q)} \qquad (11)$$

where $\boldsymbol{K_{Pos}}$ and $\boldsymbol{K_D} \in \Re^{n \times n}$ are positive-definite gain matrices; $\dot{\boldsymbol{x}}$ is the motion twist (velocity vector) of the end-effector; n is the number of joints; $\boldsymbol{g(q)}$ is the $(n \times 1)$ vector of gravity force, $\boldsymbol{q} \in \Re^n$ is the vector of joint variables, $\mathbf{J}_\theta^{\mathbf{T}}$ is the transpose of the $n \times 6$ robot Jacobian matrix, $\boldsymbol{x_d}$ and $\boldsymbol{x}$ are the desired and current pose of the robot end-effector in the Cartesian space, respectively. Assuming the quasi-static case when the joint controllers can maintain their desired set points independently and dynamic effects can be neglected, Eq. (11) can be considered as the Cartesian **inner** position controller of our robot.

In order to include the derived tactile error vector into the robot motion controller, $\boldsymbol{x}_d$ in Eq. (11) is modified so that it is defined as follows:

$$\boldsymbol{x_d} = \boldsymbol{x_a} + \boldsymbol{u} \qquad (12)$$

where $\boldsymbol{x_d}(t) = \boldsymbol{x}(t+1)$ is the desired pose, $\boldsymbol{x_a} = \boldsymbol{x}(t)$ is the current (actual) pose calculated by the Forward Kinematics of the robot, and increment $\boldsymbol{u}$ is the control disturbance – which represents the contact coordinate frame error in the Cartesian space – obtained by applying proportional-integral (PI) regulator to $\Delta\boldsymbol{s_{task}}$. PI regulator is given by a 6×6 diagonal matrix to regulate each element of the error-vector as $\Delta\boldsymbol{s_{task}} \in \Re^6$. This control disturbance is expressed in the sensing frame and, therefore, it must be transformed to the base frame of a robot arm. Thereby, the control input in the Cartesian space $\boldsymbol{u}$ is given by:

$$\boldsymbol{u} = \mathbf{Ad}_{\{\mathbf{g}\}\mathbf{T}^{\{\mathbf{s}\}}}\left(\boldsymbol{K_P}\Delta\boldsymbol{s_{task}}(\boldsymbol{t}) + \boldsymbol{K_I}\int \Delta\boldsymbol{s_{task}}(\boldsymbol{t})dt\right) \qquad (13)$$

where $\mathbf{Ad}_{\{\mathbf{g}\}\mathbf{T}^{\{\mathbf{s}\}}}$ is the adjoint matrix derived from the current forward kinematics ${}_{\{g\}}\mathbf{T}^{\{s\}} = {}_{\{g\}}\mathbf{T}^{\{e\}}\,{}_{\{e\}}\mathbf{T}^{\{s\}}$ that includes the translational plus rotational transformation from the sensor frame $\{s\}$ to the global frame $\{g\}$, passing through the end-effector frame $\{e\}$; $\boldsymbol{K_P}$ and $\boldsymbol{K_I}$ denote diagonal matrices of the proportional and integral regulator's gains, respectively.

Substituting Eq. (9) and Eq. (7) into Eq. (13) and the result into Eq. (11) leads to the following:

$$\begin{aligned}\boldsymbol{\tau_d} = \mathbf{J}_\theta^{\mathbf{T}}[\boldsymbol{K_{Pos}}\big({}_{\{g\}}\mathbf{T}^{\{s\}}(\boldsymbol{K_P}\mathbf{P}\mathbf{J}^{-\mathbf{1}}(\boldsymbol{s}_d(t) - \boldsymbol{s}(t)) + \\ \boldsymbol{K_I}\int \mathbf{P}\mathbf{J}^{-\mathbf{1}}(\boldsymbol{s}_d(t) - \boldsymbol{s}(t))dt\big) - \boldsymbol{K_D}\dot{\boldsymbol{x_a}}] + \boldsymbol{g(q)}\end{aligned} \qquad (14)$$

The PI regulator – all of the gains of the regulator are tuned manually during experiments – is added after the projection matrix in order to control how fast the tactile feature-error vector will be regulated to zero. Thus, each element in this error vector of the robot end-effector pose will be modified by its own regulator. Figure 8 shows the block diagram of the tactile servoing controller. When the end-effector interacts with the environment, the tactile sensing array provides with the information about the contact frame given by a two-dimensional distribution of pressure $I(x, y)$ from which the actual tactile features are extracted. The control cycle starts by computing the deviation $\Delta\boldsymbol{s}$ of the current tactile feature vector $\boldsymbol{s_{edge,point}}$ from the desired set of features $\boldsymbol{s_d}$. After multiplication with $\mathbf{J}^{-\mathbf{1}}$ and $\mathbf{P}$ this error is fed into the mentioned above PI controllers of Eq. (14), acting independently on all feature-error components Δs_{task} (Eq. (9)) in the Cartesian sensor frame $\{s\}$. The resulting control disturbance $\boldsymbol{u}$ is then translated and rotated from the sensor frame to the base (global) frame by applying the transformation matrix ${}_{\{g\}}\boldsymbol{T}^{\{s\}}$. This disturbance ${}_{\{g\}}\boldsymbol{u}$ will be sent as

Algorithm 1: External hybrid tactile-position control scheme

input : Current $\boldsymbol{s_a}$ and desired $\boldsymbol{s_d}$ features from tactile sensor, current robot pose $\boldsymbol{x_a}$

output : Robot end-effector command $\boldsymbol{x_d}$ for the inner position controller

1 For ($t = 0$ to t_{end} by Δt)

2 $\quad \boldsymbol{J^{-1}} \leftarrow \mathbf{J^{-1}}(\boldsymbol{s}, \boldsymbol{x_a})$ *// Tactile Jacobian computation (Eq. (8)) and update (Eq. (5))*

3 $\quad \Delta\boldsymbol{s_{edge,point}} \leftarrow \boldsymbol{s_d} - \mathbf{J^{-1}}\boldsymbol{s_a}$ *// Computation of tactile feature error (Eq. (7))*

4 $\quad \Delta\boldsymbol{s_{task}} \leftarrow \mathbf{P}\Delta\boldsymbol{s_{edge,point}}$ *// Applying the task projection matrix (Eq. (9))*

5 $\quad \boldsymbol{u} \leftarrow PI(\Delta\boldsymbol{s_{task}})$ *// Convergence of the tactile error using the PI regulators (Eq. (13))*

6 $\quad \boldsymbol{u} \leftarrow {}_{\{g\}}\boldsymbol{T}^{\{s\}}$ *// Transformation from the sensor frame to the global frame*

7 $\quad \boldsymbol{x_d} \leftarrow \boldsymbol{x_a} + \boldsymbol{u}$ *// Update of the desired end-effector pose with the tactile error (Eq. (12))*

8 $\quad$ **Send** $\boldsymbol{x_d}$ to the inner Cartesian controller

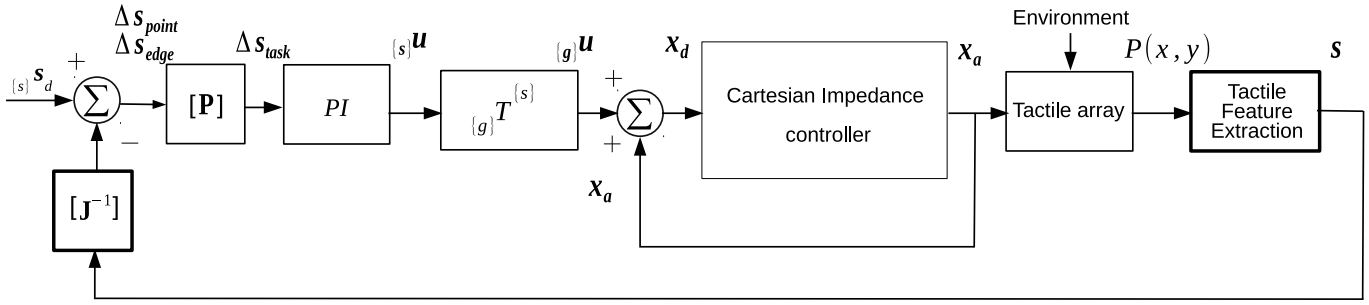

Figure 8: Block diagram of the external hybrid tactile-position controller. The Cartesian impedance controller is a part of a robot with its default stiffness and damping parameters.

the input to the inner Cartesian controller of the robot (Eq. (11)) so that the end-effector of the robot is driven to the desired pose $\boldsymbol{x_d}$.

Therefore, the proposed tactile servoing scheme is based on two different control loops: an outer loop using the tactile information and an inner loop implementing the robot motion controller. The outer loop gives the pose set points to the inner loop based on the convergence of the tactile feature error.

5. Experimental validation of the approach by case-studies

The efficacy of the proposed approach is benchmarked by performing with a robot arm equipped with a pressure sensing array the following robotic applications (case-studies): tactile exploration and and co-manipulation. First, the platform is explained. Then, in the case-studies the efficacy of the proposed method is validated. Especially, we validated the performance $DZMP$ feature and compared an error convergence with the base-line approach proposed in [7] in edge servoing task (Section 5.2.1). The usefulness of the inverse tactile Jacobian and projection matrix is validated in object exploration (Section 5.2) co-manipulation tasks (Section 5.3), respectively.

5.1. Experimental platform

Experiments are carried out using the 6 × 14 Weiss WTS0614 tactile array (size of 25 × 50 mm in total and 3.4 × 3.4 mm of each element, which are described in Appendix A) attached onto the 7 DOF KUKA LWR 4+ robot arm end-effector, the geometrical central axes of which being coincident. The origin of the sensor attached frame is set to be at the geometrical center of its sensing surface. The 7 DOF Kuka arm is operated in the Cartesian space compliance mode using the default stiffness and damping parameters of 1000 N/m and 0.7 Ns/m, respectively. The software implementation is realized via the Orocos Real-Time Toolkit (RTT) and the RTT to Robot Operating System (ROS) integration package installed on the arm PC with Linux Ubuntu 14.04 distribution patched with a Xenomai real-time development framework. The Cartesian position controller sends new commands at 1 kHz through the company's Fast Research Interface (FRI) library for the KUKA Lightweight Robot. The control law of the Cartesian motion controller represents a Cartesian virtual spring and the joint torques are computed by the Cartesian law according to Eq. (11). The gravity (and dynamic disturbances) compensation term are calculated within the KUKA robot controller (KRC). The frequencies of the control loops, the time for tactile data processing and the data acquisition are summarized in the Table 2. In the real implementation of the tactile controller (Eq. 12), the current pose of the arm was measured once at the first cycle and then at each next cycle the commanded pose was assumed to be the current pose for the future cycle in order to avoid integration of the sensors noise:

$$\boldsymbol{x}(t) = \boldsymbol{x}(t-1) + \boldsymbol{u} \tag{15}$$

where $\boldsymbol{x}(t+1)$ is the desired pose, $\boldsymbol{x}(t-1)$ is the desired pose from the previous cycle and assumed to be the current actual pose $\boldsymbol{x}_a$, $\boldsymbol{u}$ is the control disturbance that is given by contact frame error. The only difference with the theoretical one is that the current pose is not measured at each cycle, but assumed to be equal to the previously commanded pose. This assumption is valid as long as the robot reaches the desired pose at the given cycling rate.

Table 2: Rates of control loops, data acquisition/processing for the tactile servoing with the KUKA arm.

Description	Rate
Cartesian motion controller (Eq. (11))	1 kHz
Tactile servoing controller (Eq. 12).	100 Hz
Tactile image processing.	140 Hz
Tactile sensor data acquisition.	200 Hz

5.2. Exploration of unknown objects by sliding over edges: inverse tactile Jacobian validation

In the given case-study, we benchmarked the performance of $\mathbf{J^{-1}}$. Specifically, the importance of the touch-dependent entries of the Jacobian is validated by performing the task explained in the following.

In order to explore an object, as for example a metallic bar, whose location is unknown, a robot can move its

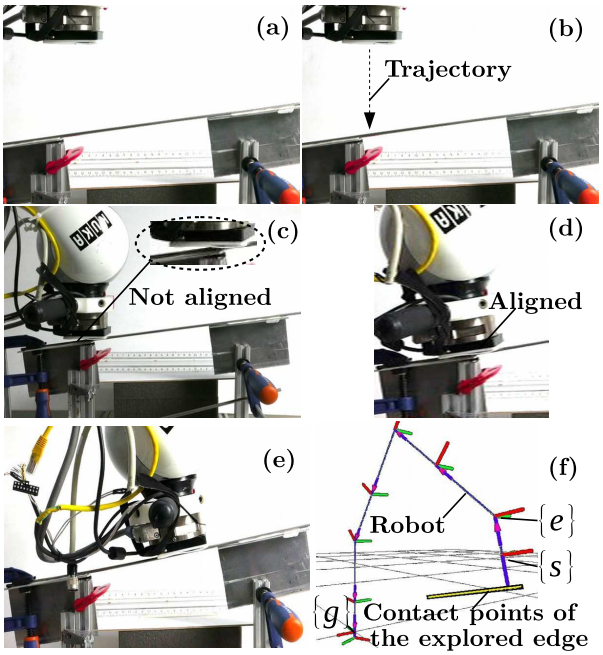

Figure 9: Exploring an unknown shape of a metal bar: align with an edge and move along the detected edge. (a) The initial state. (b) Trajectory that the end-effector follows until (c) it detects a contact. (d) Alignment with an edge. (e) Following the edge. (f) Reconstructed bar at the end of the exploration.

end-effector with a tactile array until the appearance of a contact. The contact is detected as soon as any tactel's value exceeds a threshold value, which was equal to 0.2 N (the firing up sensitivity of the sensor as depicted in Fig. A.15). Then, for the exploration of the bar, the motion of following the edge is performed by servoing the normal force, orientation of the edge, center of contact along the axis perpendicular to the edge (keeping the edge in the middle of the sensor), and the moment of force around the axis perpendicular to the edge (keeping a uniform line force distribution), thus setting $\gamma_z = 1$, $\gamma_y = 1$, $\gamma_{wx} = 1$, and $\gamma_{wz} = 1$ in $\mathbf{P}$. When the orientation of the bar is unknown, a contact area can be increased by performing rolling motion around a contact point. However, in this experiment, we assumed that the edge lies along the x-axis of the sensor frame and the edge is kept at the center of the sensor. CoP_y keeps the edge at the center of the sensing surface and the CoP_x sets in motion the end-effector to slide over the bar. The tangential forces are assumed to be rather small and not affecting sliding motions. In existing approaches [7], the bar was already in contact with the sensor and also aligned with the sensing surface. Since our tactile control architecture is flexible, it is able to implement a more general exploration strategy, where the bar is not in contact nor parallel with respect to the sensor frame at initial state (9 a).

This experiment can be divided into four different phases according to the different configurations of $\mathbf{P}$ and $\mathbf{J^{-1}}$:

- Phase I: free motion until contact with the bar

  During phase I, the robot follows an off-line generated trajectory (Fig. 10(a, b)) in order to find the contact with the bar. In our work, it was a line, starting from the actual pose of the end-effector and moving down in z direction until the contact. In Fig. 10(a), the end-effector position along z-axis is depicted with green line. To let the end-effector follow this Cartesian trajectory until the sensor touches the bar, $\mathbf{P} = 0_{6,6}$.

- Phase II: aligning (increasing the area of contact) with the bar

  At the instant of contact (Fig. 10(c)) a point-contact is detected, and the sensor frame starts aligning with the bar by rotating around the y-axis (Fig. 9(d)), which is linked to CoC_x (5th row of the Jacobian and Projection matrices). In fact, the desired CoC_x is set to 0 (Fig. 10(c)) and while the area of contact increases via this y-axis rotation, CoC_x moves towards the center of the sensor until the instant when the point contact becomes an edge contact. At the same time normal force (3rd row of the Jacobian and Projection matrices) is also controlled for keeping contact (Fig. 10(b)). CoP_y (2nd row of the Jacobian and Projection matrices) is also controlled to zero in order to keep the bar inside the area of the sensor, as near as possible to its center (Fig. 10(b)). During the increase of the area of contact via a rotation around the y-axis, CoC_x moves towards the center of the sensor until the instant when the point contact becomes edge contact (as in Fig. 4). Hence,

$$\mathbf{J^{-1}} = \begin{pmatrix} 100000 \\ 010000 \\ 001000 \\ 010000 \\ 100000 \\ 000000 \end{pmatrix}, \mathbf{P} = \begin{pmatrix} 000000 \\ 010000 \\ 001000 \\ 000000 \\ 000010 \\ 000000 \end{pmatrix} \quad (16)$$

- Phase III: switch to edge control

  When the edge is detected in the tactile image, the robot switches the mapping of the rotational motion around the y-axis, previously linked to the error in CoC_x, to the error in $DZMP_y$. In addition, CoP_y and α_z features are controlled from this event until the end of exploration. Therefore,

$$\mathbf{J^{-1}} = \begin{pmatrix} 100 & 0 & 0 & 0 \\ 010 & 0 & 0 & 0 \\ 001 & 0 & 0 & 0 \\ 010 & 0 & -1 & 0 \\ 100 & -1 & 0 & 0 \\ 000 & 0 & 0 & 1 \end{pmatrix}, \mathbf{P} = \begin{pmatrix} 000000 \\ 010000 \\ 001000 \\ 000000 \\ 000010 \\ 000001 \end{pmatrix} \quad (17)$$

  This phase is active as long as the orientation of the edge in the tactile image is not perfectly centered and aligned with the x-axis (the errors between the desired and feedback features: α_z^{des} and α_z, $DZMP_y^{des}$ and $DZMP_y$, CoP_y^{des} and CoP_y are all zero). The oscillations of the set points (Fig. 10(a)) of the end-effector pose are caused by the sensor noise. They are compensated by the damping and stiffness coefficients of the arm controller.

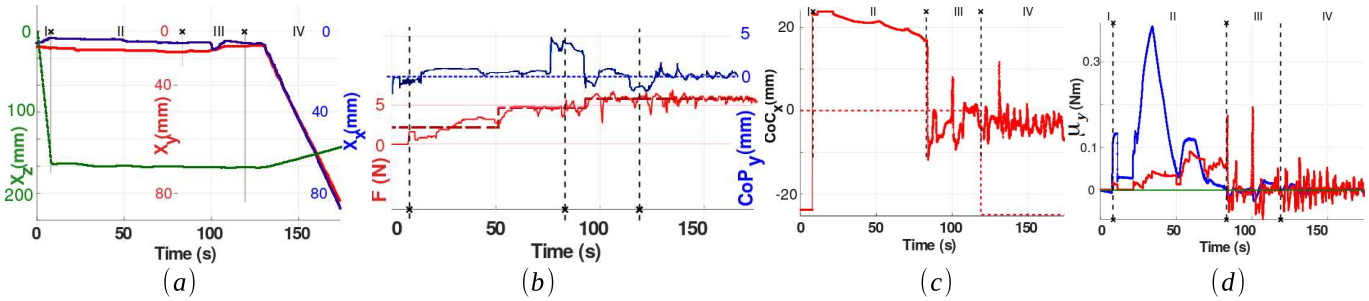

Figure 10: Plots of the evolutions of: (a) the position of the end-effector along x- (red), y- (blue), and z- (green) axes, (b) feedback (solid line) and desired (dashed line) forces (red), and feedback (solid) and desired (dashed) CoC_y (blue), (c) the feedback (solid) and desired (dashed line) CoC_x, (d) the feedback $DZMP_y$ (red) and its desired value (green) compared with ground-truth moment of force (blue).

- Phase IV: exploration

  After the complete alignment of the bar with the sensing surface in phase III, the robot starts the exploration by linear motion along x-axis. The motion is achieved by injecting in a pseudo-measurement of CoP_x ($CoP_x^{des} = -0.025$), which is depicted as the dashed line in Fig. 10(c). The feature error ΔCoP_x is linked to the translational motion along the bar. Variable elements in the projection matrix are set as follows

$$\mathbf{P} = \begin{pmatrix} 100000 \\ 010000 \\ 001000 \\ 000000 \\ 000010 \\ 000001 \end{pmatrix} \tag{18}$$

  Thereby, the projection matrix activates the translational motion (Fig. 9(e)) along the x-axis of the sensing frame in order to start a sliding motion along the bar (y- and x-axes coordinates of the end-effector depict the motion during the phase IV in Fig. 10(a)) while simultaneous control the rest of the tactile features as illustrated in Fig. 10(b). The tactile Jacobian is the same as in phase III. As a result, we recorded the cloud of contact points (the yellow dots in Fig. 9(f)) at the end of the exploration. As a result, the unknown shape of the bar can be reconstructed at the end of the exploration.

The touch dependent elements α of $\mathbf{J}^{-1}$ react to the contact type. In the previous approaches and the state-of-the-art approach [7], the elements of the tactile Jacobian are constant. Therefore, during phases II and IV, alignment and sliding over the edge could be performed only by disabling the feature errors in the corresponding directions and feeding an external motion twist.

In our approach, all of the control perturbations to the end-effector are defined in the feature space. So far, we demonstrated the efficacy of the proposed inverse tactile Jacobian not only for point and edge contacts but also for transitions between them (as shown in Phases II and III).

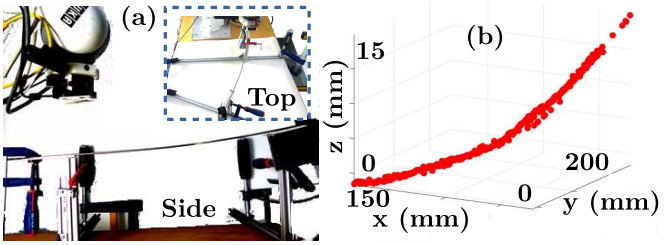

Figure 11: Exploration of a bent bar. (a) The bar and the robot from side and top views. (b) The resulting points of contact of the exploration.

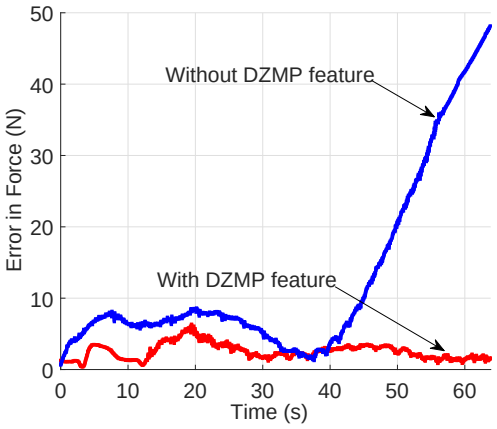

Figure 12: The comparison of the error convergence of the force during the bent (in two planes) bar exploration without (baseline approach [7] in blue) and with (proposed approach in red) use of the control of the $DZMP$ feature. The desired values were set to 5 N.

5.2.1. Comparison of the proposed approach in bent bar exploration challenge

This section describes the quantitative comparison of the proposed method with the base-line approach. An error convergence of a tactile feature is shown for a real case that has not been considered in the previous approaches.

In the first experiment, the bar was straight. However, it can be bent so that a curve appears in two planes, for example in xy and xz planes (Fig. 11(a)). Assuming that the bar is already in contact and aligned, then both $\mathbf{J}^{-1}$

and **P** are the same as in the phase IV of the above experiment. The result of the exploration is the history of the points of contact (Fig. 11(b)). Fig. 11(c) illustrates the comparison of the base-line approach [7] (depicted in blue) and the proposed in this work approach (red color): the histories of the error convergence for the force f, recorded by the ground-truth force and torque sensor. The desired force value was set to 5 N. Without the control of the $DZMP$ feature, the edge exploration task of the bent bar fails: the applied force was not compensated and raised up to 50 N, whereas it was maintained the around 5 N using our approach.

5.3. Co-manipulation: projection matrix validation

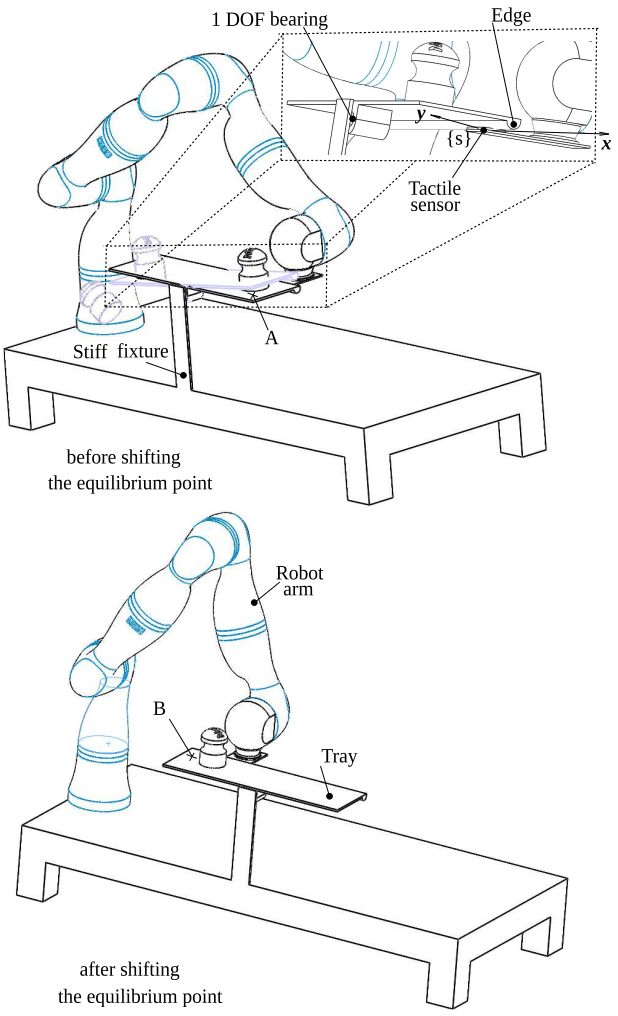

Figure 13: The platform used for balancing a 1 DOF tray in co-manipulation task. (Top) The robot arm holding the tray that can rotate around its center thanks to its 1 DOF bearing. If the object on the tray will shift from the initial equilibrium point A, the tray would rotate and the object would fall down (depicted with violet contours). (Bottom) The robot arm performed a corrective motion to keep the tray at its equilibrium state. The center of mass of the tray shifted from point A to B.

In the last case-study, we benchmarked the performance of **P**. Specifically, the importance of the task-dependent entries of the projection matrix is validated by performing the following experiment.

In this experiment, the tactile sensing array was attached onto the end-effector via an additional adapter piece (rigid 3D printed flat plate) as illustrated in Fig. 13(zoomed part). Thanks to such configuration, the robot could hold a flat tray from one side of the tray. The tactile array was attached as shown in Fig. 13(zoomed part). The opposite side could be held by a person. However, in order to avoid a human factor, the opposite side of the tray was held by a fixed wall mount via 1 DOF rotating bearing (Fig. 13(Top)). Thus, the tray could rotate around y-axis of the sensing coordinate frame (Fig. 13 (violet contour)). A copper bar was glued at the bottom side of the tray in order to enforce edge-contact type on the tactile sensor surface (Fig. 13 (zoomed part). In order to decrease friction forces to enable smooth sliding over the edge, a paper sheet was placed on top of the sensing array. The task was to maintain this tray at its equilibrium by steering the contact frame. More specifically, the goal was to keep at zero the moment of force about y-axis applied to the sensor.

The experimental scenario was as follows. The robot arm held the tray from the bottom (Fig. 13). Initially the center of mass (CoM) of the tray was at the point A. The robot arm held the tray with an object (a scale weight with mass of 200 g) so that no moment of force was applied onto the sensor. At this equilibrium state, the CoM was on the straight line connecting the attachment point of the bearing and the holding point of the end-effector (equilibrium axis). When a human operator moved the object on the tray, the CoM was shifted from the initial location to point B. In this connection, the edge of the tray applied a moment of force to the sensing surface. At this not equilibrium sate, the CoM was not on the equilibrium axis anymore. In order to keep the tray at its equilibrium, the robot arm slided along the edge by steering the contact frame using $DZMP_y$, CoC_y, f, and α_z tactile features. Specifically, the end-effector did not rotate in order to compensate the emerged moment of force, but moved along x-axis of the sensing frame based on the error $\Delta DZMP$ given $DZMP_y^{des} = 0$. This feature error was mapped to the error of the end-effector pose via the projection matrix **P** with $\gamma_{com} = 1$. Fig. 13(Top) illustrates the possible failure scenario when the object (depicted in violet) would fall down if the robot would not react to the changes of the CoM. Fig. 13(Bottom) exemplifies the new configuration of the arm when the CoM was shifted from the point A to B.

In such particular task and conditions, the elements of the inverse tactile Jacobian matrix including β

$$\mathbf{J^{-1}} = \begin{pmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \\ 0&1&0&0&-1&0 \\ 1&0&0&-1&0&0 \\ 0&0&0&0&0&1 \end{pmatrix} \tag{19}$$

Only four features are needed for the given co-manipulation task: $DZMP_y$ (to keep the equilibrium with respect to the CoM), CoC_y (to keep the contact at the center of the sensing surface), force (to keep the contact), orientation of an edge (to align the contact along the x axis of the sensing frame). These tactile feature errors are mapped to the error in contact frame by the tactile projection matrix

$$\mathbf{P} = \begin{pmatrix} 0&0&0&1&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&1 \end{pmatrix} \tag{20}$$

Hence, the contact frame error:

$$\Delta \boldsymbol{x} = [\Delta DZMP_y, \Delta COC_y, \Delta d_z, -, -, \Delta \alpha_z]^T \tag{21}$$

that drives the end-effector to the new position shown in Fig. 13(Bottom).

Fig. 14(a-d) show the evolution of features, pose of the end-effector, and the pressure profile of the sensing array during the task of keeping the CoM on the equilibrium axis. The first marker " × " indicates the instant when the object is moved from the point A, which causes the increase of $DZMP_y$ (Fig. 14(a)). The force is desired to be constant (6 N). The error of the $DZMP_y$ feature is mapped to the translational motion of the end-effector (see the changes of x- an y- coordinates of the end-effector in Fig. 14(b)) in order to compensate the appeared moment of force (the pressure distribution is not uniform as shown on left-hand side of Fig. 14(d)). At the instant when the CoM is again on the equilibrium axis (second " × " indicates this event in Fig. 14) – at this time the pressure distribution shown on the right-hand side of Fig. 14(d) is uniform –, the $DZMP_y$ converged to the desired value. It is worth noting that the orientation of the line in the tactile image caused by the edge of the tray is also controlled (Fig. 14(c)) in order to keep the orientation of the edge parallel to the x-axis of the sensor attached frame. The amount of rotation is equal to 3 degrees only, which was caused by the sensor noise.

The moment of force caused by disturbing the whole system from its equilibrium point was compensated by the translational motion activated by γ_{com} in the projection matrix. Then, in the tactile servoing control loop, this moment of force was within the feedback to regulate the appeared errors to zero (Fig. 14(a)). Such manipulation with the tactile servoing controller was not possible with projection matrices proposed in the previous works.

6. Discussions and conclusions

In order to validate our approach, we implemented different applications of the tactile servoing approach for a robot arm equipped with a pressure sensing tactile array. These applications included object exploration by following an edge of an object, manipulation by rolling back and forward, exploration by rolling, and co-manipulation tasks. The same controller is used in all cases thanks to the introduced touch dependent tactile Jacobian and projection matrix that map tactile feature errors into the robot end-effector pose error in a desired way. The developed experiments with the 7 DOF KUKA arm prove that the framework performs well in different types of manipulation and exploration tasks. The proposed features and controller can be applied for human-robot co-manipulation tasks using a robot hand interacting with the environment by means of a pressure sensing array installed on its palm. The success of the proposed framework could be extended to other tasks involving contacts and tactile feedback.

The first part of this work was to extract the information from the signals provided by a pressure sensing array. When an object presses the sensing array, a contact frame can be reconstructed using the pressure/force values at each sensing element. The contact frame is represented by tactile features. These features allowed us to reconstruct the contact frame during the physical interaction with a vertex and edge under uniform and non-uniform line forces.

With the task at hand to develop a controller for executing different contact-based tasks, we have implemented

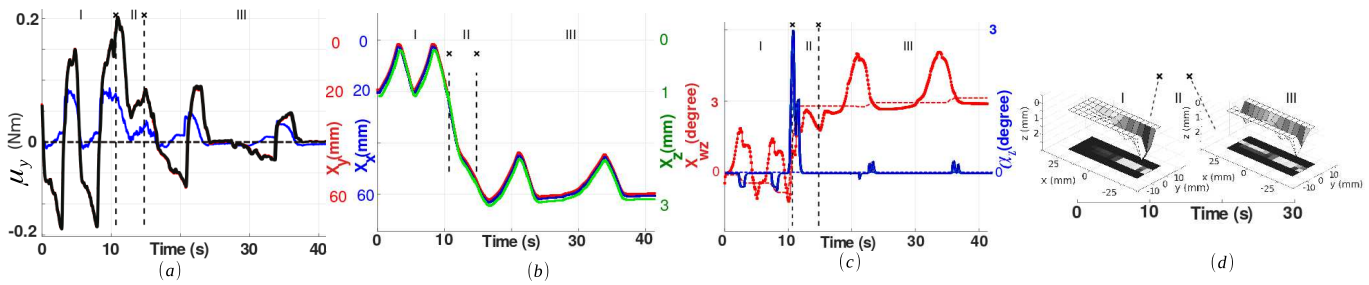

Figure 14: Evolution of features and end-effector pose coordinates during the co-manipulation task: (a) error convergence of the feedback (solid) to the desired (dashed) $DZMP_y$ feature value (black) compared with the ground-truth moment of force (blue), (b) time-sequence of x-y-z-coordinates of the end-effector's pose, (c) error convergence of the feedback (solid) to the desired (dashed) α_z feature (blue) and the corresponding set points (dashed) and real (solid) rotation of the end-effector around z-axis of the sensing frame (red), (d) gray scale representation of the pressure distribution at the non-equilibrium point of the CoM (left-hand side) and at the equilibrium point of the CoM (right-hand side).

a tactile servoing controller that uses proposed tactile features. The aim was to design a controller that would drive a robot to steer the contact frame in a desired manner. In this connection, we extended a tactile servoing approach proposed by Li et. al [7]. Their and other related surface exploration approaches, e.g. [21], have not been able to work for full 6-DOF and not able to deal with a number of edge cases, in which a robot sees only part of an edge due to misalignment or when the edge is not parallel to the sensing surface. We overcame these limitations by considering non-uniform force profiles along an edge contact. Specifically, we considered the difference of the center of contact and the center of pressure as additional tactile features ($DZMP_x$ and $DZMP_y$). In order to perform corrective sensor motions based on these tactile features, we proposed and validated the inverse tactile Jacobian and projection matrix in an edge servoing task (Section 5.2) and a co-manipulation task (Section 5.3), respectively. The inverse tactile Jacobian has two additional entries to the one introduced earlier in [5, 7]. And edge servoing task over a bent in two planes bar was conducted to compare the error convergence in force control of these methods and the proposed method. The proposed tactile based approach succeeded in the mentioned task, whereas the error did not converge to zero for the state-of-the-art control algorithm. A more rigorous analysis of the controller, including frequency response, is omitted as every case is different, especially the case with the exploration of the floating bar. However, we elaborated geometrical consistency and relied on the stability of classical PI regulators and the natural and artificial constraints analysis shown in Appendix C.

There are two main limitations of the proposed approach. First, since we considered planar tactile sensor arrays only, the feature extraction methods described in this paper will not work for non - planar tactile sensors and for small surface structures. Mapping must be defined for the sensors with specific shapes, as for example the ones that have a human fingertip shape. Secondly, as soon as the contact disappears the tactile controller does not send new set points to the robot. Due to the Cartesian impedance parameters of the robot manipulator and the compliance of the sensing surface, this problem was mitigated. However, in the line tracking experiments with bent bars, the exploration (sliding) velocity was limited since at a rather high sliding velocities the end-effector could not align with the edge in time and then the contact was lost (a time scale theory applied to a hybrid system [30] can be considered to mitigate this limitation). Another issue with higher sliding velocities is the violation of the quasi-static assumption when we could approximate the non-linear response of the sensing cells (Fig. A.15) with linear compliance constants $\boldsymbol{C}$ defined in Eq. 2. For the same reason, in the experiment described in Sec. 5.2, we set the gains of regulators (Eq. 13) so that it took around 100 seconds for aligning with an edge (Fig. 10(a)). Nevertheless, this time can be significantly decreased by finely tuning the gains.

For the future work, we envision to extend our approach to a more general one, in which a task and robot with its abilities would be jointly considered by including information from multiple tactile sensors distributed over different links, which was partially fulfilled in robot hand control [22].

7. Acknowledgments

The authors are very thankful to Vincent Hayward for discussion on tactile perception, and Guillaume Walck and Vincent Padois for discussions on technical side of implementations.

Appendix A. Tactile sensor

The 6 × 14 (size of 25 × 50 mm in total and 3.4 × 3.4 mm of each element) Weiss Robotics WTS0614 piezoresistive tactile sensor is used in the current work. The sensor is covered with a layer of an elastic material – electro conductive rubber. Its output force responses were identified using the following tools: a schematic drawing of the elements' locations, the ground-truth Ati Nano17 force and torque, a custom made cylindrical indenter with 1 mm diameter, the proxxon mf70 D-54518 3 axes milling device instead of an X-Y-Z manipulator. The ATI Nano17 force sensor was attached onto the head of the 3 axes manipulator. The indenter printed using ABS plastic was attached onto the force sensor. The indenter pushed against the elastic sensing surface over the centers of tactels. The deformations in z direction of the sensor, tactile sensor outputs, and the force sensor measurements were captured starting at the surface up to the saturation of the sensor outputs by pressing the elastic surface incrementally. Fig. A.15 shows the sensor responses versus the ground-truth force measurements. The linearized response is depicted with the black color, though the responses can be characterized with a higher order polynomial (red color). The maximum pressure applied by the indenter with diameter of 1 mm under the constant force is equal to (2 N/ 0.785 m^2)10^6 = $2.548 \cdot 10^6$ Pa ($F_{max}/(\pi \cdot r^2)$), that is 0.25 N/mm^2 or 2.89 N/taxel. There is a significant difference in the sensitivity of the tactel 15, which is close to the center, and tactel 1, which is located at the boarder. These irregularities are the side effects of the construction of the sensor: the rubber at the corners is not as flexible as at the center because the rubber is attached to the base of the sensor from the sides.

Appendix B. Image moments and PCA

$$M_{i,j} = \sum_{x=1}^{N_x} \sum_{y=1}^{N_y} x^i y^j I(x, y) \tag{B.1}$$

where $M_{i,j}$ is the i, j-th order image moment; N_x and N_y are the number of pixels along x and y axes, respectively;

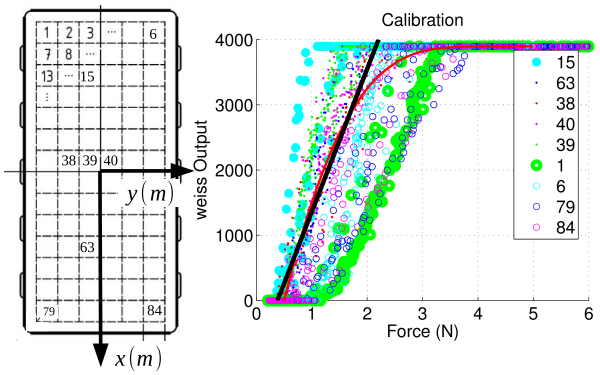

Figure A.15: Responses of the individual sensor elements to the applied forces on each element. The sensor outputs are the raw values that are proportional to the (resistance) voltage: the WEISS WTS 0614 sensors use piezoresistive technology.

$I(x, y)$ is the intensity of x, y pixel. When $i + j = 0$, the order of the image moment is zero. When the sum is equal to 1, the moment is the first order and so on. center of pressure is given by

$$CoP_x = \frac{M_{1,0}}{M_{0,0}} \tag{B.2}$$

where $M_{1,0}$ is the first moment with respect to x-axis, $M_{0,0}$ is the zeroth moment.

$$CoP_y = \frac{M_{0,1}}{M_{0,0}} \tag{B.3}$$

where $M_{0,1}$ is the first moment with respect to y-axis.

The orientation of the edge is given by the principle component representing the direction, in which the contact points are spread the most. This principal component U_p is equal to the eigenvector with the highest eigenvalue of the covariance matrix Q:

$$Q(x, y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n - 1} \tag{B.4}$$

where $n = 84$ for 6×14 tactile image $I(x, y)$, $\bar{x}$ and $\bar{y}$ are the mean values of x and y. The angle of rotation of the base vector of U in x direction (U_1) around $z-$ axis is given by:

$$\alpha_z = atan2(U_1, e_x) \tag{B.5}$$

where e_x is the base vector in x direction of the sensor frame.

Appendix C. Geometric consistency: natural and artificial constraints

The convergence of tactile feature error is achieved via the external PI regulators and the internal PD and gravity compensation motion control. The robustness of the controller depends on the geometric consistency (constraints compatibility) since there exist different kinematic constraints that can change during physical interactions with objects.

Following the decomposition approach [26], we set the artificial constraints on the forces and velocities separately. However, not all of the directions in the Cartesian space can be artificially constrained when there is an interaction of a robot with a rigid environment. Since the robot end-effector can not move freely in some directions due to the contact with the environment, there are natural constraints on motions along these directions. There are also directions in which forces are naturally constrained. These constraints can vary depending on the type of contact and task. These natural constraints change for point- and edge- contact types. Therefore, artificial constraints obey to the changes in the natural constraints thanks to the interaction matrix and the introduced touch depended inverse tactile Jacobian.

The decomposition between the interaction forces and possible motions is as followed:

$$\boldsymbol{v}^T \boldsymbol{f} = \upsilon_x f_x + \upsilon_y f_y + \upsilon_z f_z + w_x \mu_x + w_y \mu_y + w_z \mu_z \tag{C.1}$$

where the υ and w are the translational and rotational velocities (twist), f and μ are the forces and moments (wrench). When the forces and velocities are decomposed, it is said that a **reciprocity condition** is satisfied. The condition is following:

$$\boldsymbol{v}^T \boldsymbol{f} = 0. \tag{C.2}$$

Table C.3 summarizes the configurations of the natural and artificial constraints, satisfying this condition, during the interactions using a planar sensing array.

Table C.3: Natural and artificial constraints during the interactions.

Contact type	Natural constraint	Artificial constraint	Reciprocity condition($\boldsymbol{v}^T \boldsymbol{f}$)
Point	$f_x = 0$ $f_y = 0$ $\upsilon_z = 0$ $\mu_x = 0$ $\mu_y = 0$ $\mu_z = w_z = 0$	$\upsilon_x \sim \Delta CoC_x$ $\upsilon_y \sim \Delta CoC_y$ $f_z \sim \Delta force$ $w_x \sim \Delta CoC_y$ $w_y \sim \Delta CoC_x$ –	0
Edge (along x axis)	$f_x = 0$ $f_y = 0$ $\upsilon_z = 0$ $\mu_x = 0$ $w_y = 0$ $\mu_z = 0$	$\upsilon_x \sim \Delta CoC_x$ $\upsilon_y \sim \Delta CoC_y$ $f_z \sim \Delta force$ – $\mu_y \sim \Delta DZMP_y$ $w_z \sim \Delta \alpha_z$	0

[1] R. S. Johansson, J. R. Flanagan, Coding and use of tactile signals from the fingertips in object manipulation tasks, Nature Reviews Neuroscience 10 (5) (2009) 345–359. doi:10.1038/nrn2621.

[2] B. Gates, A robot in every home, Scientific American 296 (1) (2007) 58–65.

[3] Z. Kappassov, J.-A. Corrales, V. Perdereau, Tactile sensing in dexterous robot hands — Review, Robotics and Autonomous Systemsdoi:10.1016/j.robot.2015.07.015.

[4] M. Briot, et al., The utilization of an artificial skin sensor for the identification of solid objects, in: 9th Int. Symp. on Industrial Robots, 1979, pp. 13–15.

[5] H. Zhang, N. Chen, Control of contact via tactile sensing, Robotics and Automation, IEEE Transactions on 16 (5) (2000) 482–495. doi:10.1109/70.880799.

[6] A. Drimus, G. Kootstra, A. Bilberg, D. Kragic, Design of a flexible tactile sensor for classification of rigid and deformable objects, Robotics and Autonomous Systems 62 (1) (2014) 3 – 15, new Boundaries of Robotics. doi:dx.doi.org/10.1016/j.robot.2012.07.021.

[7] Q. Li, C. Schürmann, R. Haschke, H. J. Ritter, A control framework for tactile servoing, in: Robotics: Science and Systems IX, Technische Universität Berlin, Berlin, Germany, June 24 - June 28, 2013. doi:10.15607/RSS.2013.IX.045.

[8] R. S. Fearing, J. M. Hollerbach, Basic solid mechanics for tactile sensing, The International journal of robotics research 4 (3) (1985) 40–54.

[9] V. A. Ho, T. Nagatani, A. Noda, S. Hirai, What can be inferred from a tactile arrayed sensor in autonomous in-hand manipulation?, in: Automation Science and Engineering (CASE), 2012 IEEE International Conference on, 2012, pp. 461–468. doi:10.1109/CoASE.2012.6386384.

[10] R. Howe, Tactile sensing and control of robotic manipualtion, Journal of Advanced Robotic 8 (3) (1994) 245–261.

[11] M. R. Cutkosky, J. Ulmen, Dynamic Tactile Sensing, Springer International Publishing, Cham, 2014, pp. 389–403. doi:10.1177/0278364914564703.

[12] P. J. Kyberd, P. H. Chappell, Object-slip detection during manipulation using a derived force vector, Mechatronics 2 (1) (1992) 1 – 13.

[13] J. Platkiewicz, H. Lipson, V. Hayward, Haptic Edge Detection Through Shear, Scientific Reports 6 (2016) 23551. doi:10.1038/srep23551.

[14] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. E. Giannaccini, J. Rossiter, N. F. Lepora, The tactip family: Soft optical tactile sensors with 3d-printed biomimetic morphologies, Soft Robotics 5 (2) (2018) 216–227. doi:10.1089/soro.2017.0052.

[15] K. J. Overton, T. Williams, Tactile sensation for robots, in: Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'81, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1981, pp. 791–795.

[16] C. Muthukrishnan, D. Smith, D. Myers, J. Rebman, A. Koivo, Edge detection in tactile images, in: Robotics and Automation. Proceedings. 1987 IEEE International Conference on, Vol. 4, 1987, pp. 1500–1505. doi:10.1109/ROBOT.1987.1088067.

[17] A. D. Berger, P. K. Khosla, Using tactile data for real-time feedback, The International Journal of Robotics Research 10 (2) (1991) 88–102. doi:10.1177/027836499101000202.

[18] N. Chen, H. Zhang, R. Rink, Edge tracking using tactile servo, in: Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on, Vol. 2, IEEE, 1995, pp. 84–89.

[19] B. Espiau, F. Chaumette, P. Rives, A new approach to visual servoing in robotics, IEEE Transactions on Robotics and Automation 8 (3) (1992) 313–326. doi:10.1109/70.143350.

[20] P. K. Allen, Integrating vision and touch for object recognition tasks, The International Journal of Robotics Research 7 (6) (1988) 15–33. doi:10.1177/027836498800700603.

[21] H. Liu, K. C. Nguyen, V. Perdereau, J. Bimbo, J. Back, M. Godden, L. D. Seneviratne, K. Althoefer, Finger contact sensing and the application in dexterous hand manipulation, Autonomous Robots 39 (1) (2015) 25–41. doi:10.1007/s10514-015-9425-4.

[22] N. Sommer, A. Billard, Multi-contact haptic exploration and grasping with tactile sensors, Robotics and Autonomous Systems 85 (2016) 48–61. doi:10.1016/j.robot.2016.08.007.

[23] U. Martinez-Hernandez, T. J. Dodd, M. H. Evans, T. J. Prescott, N. F. Lepora, Active sensorimotor control for tactile exploration, Robotics and Autonomous Systems 87 (2017) 15 – 27. doi:https://doi.org/10.1016/j.robot.2016.09.014.

[24] N. F. Lepora, K. Aquilina, L. Cramphorn, Exploratory tactile servoing with active touch, IEEE Robotics and Automation Letters 2 (2) (2017) 1156–1163. doi:10.1109/LRA.2017.2662071.

[25] P. K. Allen, Integrating vision and touch for object recognition tasks, The International Journal of Robotics Research 7 (6) (1988) 15–33.

[26] M. H. Raibert, J. J. Craig, Hybrid position/force control of manipulators, Journal of Dynamic Systems, Measurement, and Control 103 (2) (1981) 126–133.

[27] P. Sikka, H. Zhang, S. Sutphen, Tactile servo: Control of touch-driven robot motion, in: Experimental Robotics III, Springer, 1994, pp. 219–233.

[28] A. D. Berger, On using a tactile sensor for real-time feature extraction, Ph.D. thesis, Carnegie-Mellon University (1988).

[29] B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer, 2008. doi:10.1007/978-3-540-30301-5.

[30] F. Taousser, M. Defoort, M. Djemai, Consensus for linear multi-agent system with intermittent information transmissions using the time-scale theory, International Journal of Control 89 (1) (2016) 210–220. doi:10.1080/00207179.2015.1065544.