Towards Robot Interaction Autonomy: Explore, Identify, and Interact

Pietro Balatti[1,3], Dimitrios Kanoulas[2], Nikos G. Tsagarakis[2], and Arash Ajoudani[1]

***Abstract*— Nowadays, robots are expected to enter in various application scenarios and interact with unknown and dynamically changing environments. This highlights the need for creating autonomous robot behaviours to explore such environments, identify their characteristics and adapt, and build knowledge for future interactions. To respond to this need, in this paper we present a novel framework that integrates multiple components to achieve a context-aware and adaptive interaction between the robot and uncertain environments. The core of this framework is a novel self-tuning impedance controller that regulates robot quasi-static parameters, i.e., stiffness and damping, based on the robot sensory data and vision. The tuning of the parameters is achieved only in the direction(s) of interaction or movement, by distinguishing expected interactions from external disturbances. A vision module is developed to recognize the environmental characteristics and to associate them to the previously/newly identified interaction parameters, with the robot always being able to adapt to the new changes or unexpected situations. This enables a faster robot adaptability, starting from better initial interaction parameters. The framework is evaluated experimentally in an agricultural task, where the robot effectively interacts with various deformable environments.**

I. Introduction

In the classical concept of industrial robotics, robots are deployed in structured environments of manufacturing plants, operating inside industrial cages. In this context, they are usually pre-programmed since they are aware of all their workspace features and properties. These characteristics are not changing over time, and therefore the required level of adaptation is trivial. Nowadays, robots are going beyond this outdated concept, and they are making their way into new unstructured application fields such as agriculture, disaster scenarios, small-batch manufacturing, healthcare, construction, and entertainment. These sectors demand an efficient interaction of the robots with human beings and unknown environments. The high uncertainty levels, that characterize tasks in such situations, encourage the development of context-aware and adaptive robotic behaviors able to distinguish expected interactions from external disturbances.

An intuitive way to develop adaptive robot skills is based on the observations from human demonstrations [1]–[3]. However, the limit of such approaches is given by their high dependency on the training data sets quality. Moreover, while performing complex manipulation tasks, accurate measurements of the contact forces may not be possible through wearable sensory systems, that is why most learning by demonstration techniques function on a kinematic level.

[1]HRI[2] Lab and [2]HHCM Lab, Department of Advanced Robotics, Istituto Italiano di Tecnologia, Genoa, Italy. pietro.balatti@iit.it
[3]Dept. of Information Engineering, University of Pisa, Italy.

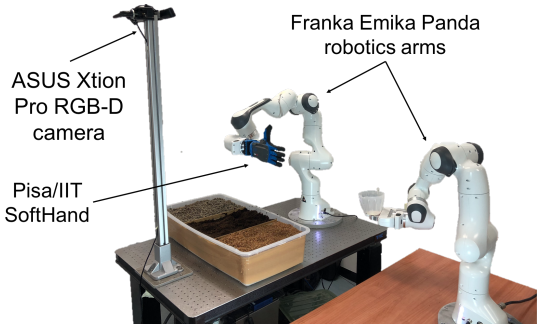

Fig. 1: The algorithm enables robotic arms to explore the material in the workspace identifying and self-tuning its impedance parameters only along the directions of interaction in an agricultural setup.

In order to solve this problem, analytical solutions have focused on the use of impedance control [4]–[6], force control [7], or hybrid interaction controllers [8], [9] to address the uncertainty levels. However, in the majority of cases, the robot programmers choose the control parameters in line with their experience in carrying out analogous tasks. By presetting these parameters, the framework cannot adapt when task conditions change, so the applicability to the control of complex interactions, and in general the adaptation capacity to unpredicted events, is very limited [10], [11].

Adaptive learning techniques to address this issue have also been proposed. In [12], an adaptive impedance controller for upper-limb rehabilitation, based on evolutionary dynamic fuzzy neural network, is proposed. In this work, the desired impedance between robot and impaired limb can be regulated. However, this method lacks of versatility, since it is limited to a specific task. In [13], empirical constants need to be set, reducing the flexibility of the framework, and the desired impedance matrices are assumed to be diagonal, resulting in limited adaptability in selective Cartesian axes. More generic methods, not task-dependent, have been introduced by reducing impedance control to position control (through high position loop gains) when there is no interaction [14] with the aim to minimize the error between the desired and actual trajectories [15]. Nevertheless, these systems respond in the same way to all interactions coming from the external world (see also [16]), as human applied forces or external disturbances, and their validity has mostly been proven in simulation environments.

To address these challenges, we propose a novel manipulation framework that integrates various components to achieve a context-aware and adaptive robot interaction behavior. The core of our framework is a novel self-tuning impedance controller, to enable robot adaptation on directions of interaction, while setting a compliant profile along all the other directions. This controller is based on our introductory work

proposed in [17], where preliminary results were shown only along major Cartesian axes in quasi-static conditions. Here, the controller is extended to tune the major axis of the Cartesian stiffness and damping ellipsoids in any arbitrary direction, which coincide with the direction of interaction.

The novel framework additionally enables a robot to *explore* an environment, to *identify* its characteristics, and to effectively *interact* with it. A new visual perception module is developed to localize different materials in the robot workspace and associate their newly/previously identified characteristics to the robot interaction knowledge (i.e., impedance control and self-tuning gains). This is similar to the behavior that can be observed in humans: we constantly build internal models of the external world, by exploring and identifying it. When interacting with new or similar environments, we consider the prior knowledge while keeping open the possibility of adaption, to update our internal knowledge [18]. Another similarity between our method and human behavior is given by the default compliant behavior of the robot. In fact, when no interaction is expected, we tend to relax our muscles to comply with unexpected external disturbances (and to minimize energy consumption).

The coordination of the framework components and their interaction is achieved through an improved Finite State Machine (FSM), which combines the data received from the robot sensors with the vision module, to decide where the robot should operate in its workspace and when to activate the self-regulating impedance, by distinguishing an expected interaction from external disturbances.

We perform proof-of-concept experiments for an agricultural setup, demonstrating the potential of the presented methodology in one of the most promising fields in robotics. We used two Franka Emika Panda robotic arms: one, equipped with the Pisa/IIT SoftHand [19] to be able to grasp two different tools firmly and reliably, and the other one to cooperate with the first Panda robot to accomplish the designed task. Fig. 1 illustrates the experimental setup.

II. Method

The proposed framework aims to equip robots with an original set of skills to explore various environments, adapt to their conditions, build the knowledge, and use it for future interactions. The concept of self-adaptability is at the core of this methodology, even after building the knowledge on task or environments. This implies that if environmental conditions are subject to variations, the robot is still able to adapt, starting from a reasonable initial condition.

The required theoretical and technological components to build such a framework are integrated into five main modules, as shown in Fig. 2: (1) a Cartesian impedance controller whose parameters can be changed online, (2) a self-tuning impedance unit responsible for modifying the aforementioned parameters when a interaction with the environment is predicted, (3) a trajectory planner that, given an initial and a target pose, calculates the spatial points to be reached by the controller, (4) a visual perception module that locates the materials' positions in the robot workspace,

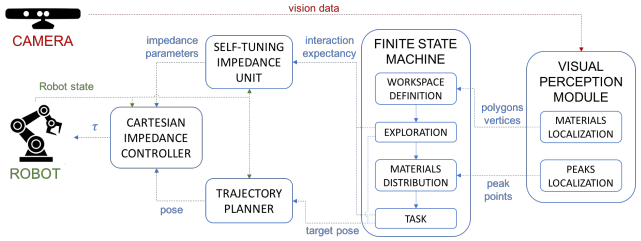

Fig. 2: The presented framework subdivided in modules. Each unit is a ROS node and data are exchanged via ROS messages on the depicted ROS topics (dotted lines). The messages published on the blue topics are generated by the designed software architecture, the one on the green topic by the Robot and the one on the red topic by the Camera.

and (5) a Finite State Machine (FSM) that, based on the data provided by (4), triggers unit (2) and (3).

A. Cartesian impedance controller

Cartesian impedance control techniques provide the ability to achieve any arbitrary quasi-static behavior at the robot end-effector [9], [20]. This is however limited to the positive definiteness and symmetry of the impedance matrices, by considering robot torque boundaries and the number of degrees of freedom (≥ 6).

This control technique relies on torque sensing and actuation, with the vector of robot joint torques $\boldsymbol{\tau} \in \mathbb{R}^n$ calculated as follows:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{\tau}_{ext}, \tag{1}$$

$$\boldsymbol{\tau}_{ext} = \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{F}_c + \boldsymbol{\tau}_{st}, \tag{2}$$

where n is the number of joints, $\boldsymbol{q} \in \mathbb{R}^n$ is the joint angles vector, $\boldsymbol{J} \in \mathbb{R}^{6\times n}$ is the robot arm Jacobian matrix, $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is the mass matrix, $\boldsymbol{C} \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal matrix, $\boldsymbol{g} \in \mathbb{R}^n$ is the gravity vector and $\boldsymbol{\tau}_{ext}$ is the external torque vector. $\boldsymbol{F}_c$ represents the forces vector in the Cartesian space and $\boldsymbol{\tau_{st}}$ the second task torques projected onto the null-space of $\boldsymbol{J}$.

Forces $\boldsymbol{F}_c \in \mathbb{R}^6$ are calculated as follows:

$$\boldsymbol{F}_c = \boldsymbol{K}_c(\boldsymbol{X}_d - \boldsymbol{X}_a) + \boldsymbol{D}_c(\dot{\boldsymbol{X}}_d - \dot{\boldsymbol{X}}_a), \tag{3}$$

where $\boldsymbol{K}_c \in \mathbb{R}^{6\times 6}$ and $\boldsymbol{D}_c \in \mathbb{R}^{6\times 6}$ represent respectively the Cartesian stiffness and damping matrix, $\boldsymbol{X}_d$ and $\boldsymbol{X}_a \in \mathbb{R}^6$ the Cartesian desired and actual position, $\dot{\boldsymbol{X}}_d$ and $\dot{\boldsymbol{X}}_a \in \mathbb{R}^6$ their corresponding velocity profiles. The Cartesian desired position and velocity are given in input by the *Trajectory planner* (see Sec. II-C).

B. Self-tuning impedance unit

Robots are expected to work with humans, and within spaces designed and built for them. Often in these kind of environments the level of uncertainty is high, where adaptive tuning of the robot impedance parameters can lead to better interaction performances. Some preliminary work in this direction has already been investigated adapting the impedance parameters based on human observations [10] and energy tank-based optimization [11], to improve the physical interaction performance based on the varying task conditions.

Such techniques, however, are not capable of distinguishing external disturbances from expected interactions, and do not have the capacity to learn from explorations for future interaction purposes.

To address these issues, we develop a novel self-tuning impedance controller that is able to adapt its parameters when an interaction is expected. The adaptation is limited to the expected direction(s) of interaction/movement to avoid unnecessary stiffening/complying of the remaining axes.

To determine when an interaction is to happen, we define a Boolean value, named Interaction expectancy ($\boldsymbol{I}_e$), that results from the Boolean logic rule $\boldsymbol{I}_e = \boldsymbol{I}_m \wedge \boldsymbol{I}_f$. $\boldsymbol{I}_m$ is set by the FSM and it is True only in the states in which an interaction with the environment is expected. $\boldsymbol{I}_f$ is given by the vision module. It is True when the tool attached to the end-effector is inside the material that has to be manipulated. The importance of this consideration has been shown in our preliminary work [17] and it will not be repeated here.

When $\boldsymbol{I}_e$ is False, the Cartesian stiffness matrix $\boldsymbol{K}_c$ is set to the default compliant behavior, i.e. a diagonal matrix with all the non-zero coefficients set to $\boldsymbol{k}_{min}$. In fact, the first rule of the proposed self-tuning impedance controller is to achieve a compliant behaviour in all Cartesian and redundant axes, unless an interaction is about to occur. This compliant behaviour can be chosen based on a trade-off between the position tracking accuracy (affected by the existence of unmodeled dynamics such as friction) and the force response, if an unexpected interaction occurs.

The damping matrix $\boldsymbol{D_c}$ is derived from $\boldsymbol{K}_c$ by:

$$\boldsymbol{D_c} = \boldsymbol{\Lambda}_* \boldsymbol{D}_{diag} \boldsymbol{K}_{adj*} + \boldsymbol{K}_{adj*} \boldsymbol{D}_{diag} \boldsymbol{\Lambda}_*, \tag{4}$$

where $\boldsymbol{D}_{diag}$ is the diagonal matrix containing the damping factor ($\zeta = 0.7$), $\boldsymbol{K}_{adj*}\boldsymbol{K}_{adj*} = \boldsymbol{K}_c$ and $\boldsymbol{\Lambda}_*\boldsymbol{\Lambda}_* = \boldsymbol{\Lambda}$, where $\boldsymbol{\Lambda}$ is the desired end-effector mass matrix [21].

On the contrary, if the interaction expectancy value is True, the Cartesian stiffness matrix $\boldsymbol{K}_c$ and consequently the damping matrix $\boldsymbol{D}_c$ are subject to changes increasing (or decreasing) the impedance parameters only along the direction of the desired movement defined by $\overrightarrow{\boldsymbol{P}} = \boldsymbol{X_{d,t}} - \boldsymbol{X_{d,t-1}}$, (which can also be calculated from $\dot{X}_d$) and keeping a compliant behavior, set to $\boldsymbol{k}_{min}$ and $\boldsymbol{d}_{min} = 2\zeta\sqrt{\boldsymbol{k}_{min}}$ [21], along the other axes. To achieve this, the stiffness and damping matrices, as being symmetric and positive definite, can be expressed by $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*$, which is known as the Singular Value Decomposition (SVD). Such a decomposition enables us to project the desired stiffness and damping, calculated w.r.t. the reference frame of desired motion vector $\overrightarrow{\boldsymbol{P}}$, onto the reference frame of the robot base. $\boldsymbol{U} \in \mathbb{R}^3$ and $\boldsymbol{V} \in \mathbb{R}^3$ are orthonormal bases, and $\boldsymbol{\Sigma} \in \mathbb{R}^3$ is a diagonal matrix whose elements are the singular values of matrix $\boldsymbol{A}$ sorted in decreasing order and representing the principal axes amplitudes of the resulting geometric ellipsoid. The columns of matrix $\boldsymbol{U}$ form a set of orthonormal vectors, which can be regarded as basis vectors. In this work, the first column of $\boldsymbol{U}$ represents the desired motion vector $\overrightarrow{\boldsymbol{P}}$, while the second and the third ones are derived from the first in such a way they form an orthonormal basis. Since the Hermitian transpose $\boldsymbol{V}^* \in \mathbb{R}^3$, and the resulting matrix $\boldsymbol{A}$, that represents the impedance values, is positive definite, we have that $\boldsymbol{V}^* = \boldsymbol{V}^T$ and $\boldsymbol{V} = \boldsymbol{U}$. Combining the previous equations, we can derive the stiffness and the damping matrices:

$$\boldsymbol{K}_c = \boldsymbol{U}\boldsymbol{\Sigma}_k\boldsymbol{U}^T,\ \boldsymbol{D}_c = \boldsymbol{U}\boldsymbol{\Sigma}_d\boldsymbol{U}^T, \tag{5}$$

where the diagonal matrix $\boldsymbol{\Sigma}_k$ and $\boldsymbol{\Sigma}_d$ coefficients are respectively the desired stiffness and damping coefficients along the direction of the vectors composing the $\boldsymbol{U}$ basis. They are diagonal matrices defined by:

$$\boldsymbol{\Sigma}_k = diag(\boldsymbol{k}_{st}, \boldsymbol{k}_{min}, \boldsymbol{k}_{min}),\ \boldsymbol{\Sigma}_d = diag(\boldsymbol{d}_{st}, \boldsymbol{d}_{min}, \boldsymbol{d}_{min}), \tag{6}$$

where $\boldsymbol{k}_{st}$ is the self-tuning stiffness coefficient to be set along the motion vector $\overrightarrow{\boldsymbol{P}}$ and $\boldsymbol{d}_{st}$ its correspondent damping element. $\boldsymbol{k}_{st}$ is defined at every time t as:

$$\boldsymbol{k}_{st,t} = \boldsymbol{k}_{st,t-1} + \boldsymbol{\alpha}\boldsymbol{\Delta P}\boldsymbol{\Delta T}, \tag{7}$$

where $\boldsymbol{\alpha}$ is the update parameter, $\boldsymbol{\Delta P} = |\boldsymbol{\Delta X} \cdot \widehat{\boldsymbol{P}}|$ is the absolute value of the Cartesian error $\boldsymbol{\Delta X} = \boldsymbol{X_d} - \boldsymbol{X_a}$ projected onto the direction of the motion vector $\overrightarrow{\boldsymbol{P}}$, and normalized $\widehat{\boldsymbol{P}}$, and $\boldsymbol{\Delta T}$ is the control loop sample time. Note that, $\boldsymbol{k}_{st}$ is subject to changes only when $\boldsymbol{\Delta P}$ is above a threshold defined as $\boldsymbol{\Delta P}_t$.

A good choice of $\boldsymbol{\alpha}$ is fundamental to reach the desired impedance in a short time to enhance robot autonomy. This is because a higher $\boldsymbol{\alpha}$ value is suitable for a fast convergence in dense material. Nevertheless, choosing a high value for non-dense material will cause unnecessary stiffening of the robot that must be avoided. Hence, we performed experiments on different materials such as soil, sand, rocks, air and water to obtain an average alpha value, as a trade-off between fast convergence and stiffening performance. To achieve this, for every material m, $\boldsymbol{\alpha}_m$ was calculated and the average value was defined through the arithmetic mean of all the n materials taken into account in the analysis.

In certain cases, the impedance parameters need to be decreased and (7) cannot be applied. For instance, when the tool attached to the robot end-effector exits the material, even still being inside the interaction expectancy area. We define $\boldsymbol{\Delta F}_{ext,t}$ as the variation of the external forces detected at the robot end-effector at time t w.r.t. the ones measured at the previous time step, i.e. $\boldsymbol{\Delta F}_{ext,t} = \boldsymbol{F}_{ext,t} - \boldsymbol{F}_{ext,t-1}$. In the aforementioned situations, $\boldsymbol{\Delta F}_{ext,t}$ is positive and $\boldsymbol{k}_{st}$ is defined at every time t as:

$$\boldsymbol{k}_{st,t} = \boldsymbol{k}_{st,t-1} - \boldsymbol{\beta}\boldsymbol{\Delta F}_{ext,t}\boldsymbol{\Delta T}, \tag{8}$$

where $\boldsymbol{\beta}$ is given by $\boldsymbol{\alpha}$ scaled by a factor of 10^{-2}, to implement a similar rate of adaptation as in (7). A pseudo-code of the proposed method is presented in Alg. 1.

C. Trajectory planner

The third component of the system is responsible for the motion planning, based on the type of trajectory received by the FSM (see Sec. II-E). It offers three different trajectory planners, i.e., *point-to-point motion*, *scooping motion*, and *shaking motion*, based on the target pose (given by the FSM) and incorporating a fifth-order polynomial.

Algorithm 1 Self-tuning impedance algorithm

Input: $\overrightarrow{P}$
Output: K_c, D_c
Initialization :
$\Sigma_k = k_{min} I_{3x3}, \ \Sigma_d = d_{min} I_{3x3}$
$k_{st,0} = k_{min}, \ d_{st,0} = d_{min}$
Control loop :
if $(interaction\ expectancy)$ **then**
$\widehat{P} = \overrightarrow{P} \ / \ \| \overrightarrow{P} \|_2$
$\Delta P = |(X_d - X_a) \cdot \widehat{P}|$
if $(\Delta P \geq \Delta P_t)$ **then**
$k_{st,t} = k_{st,t-1} + \alpha \Delta P \Delta T$
$\Sigma_{k,1,1} = k_{st,t}, \ \Sigma_{d,1,1} = 2\zeta\sqrt{k_{st,t}}$
$U = getOrthonormalBasis(\overrightarrow{P})$
$K_c = U \Sigma_k U^T, \ D_c = U \Sigma_d U^T$
else if $(\Delta F_{ext} \geq 0)$ **then**
$k_{st,t} = k_{st,t-1} - \beta \Delta F_{ext,t} \Delta T$
end if
else
$K_c = k_{min} I_{3x3}, \ D_c = d_{min} I_{3x3}$
end if
return K_c, D_c

D. Visual perception module

The visual perception module is a key element of the system, providing information to the FSM about the materials to be manipulated by the robotic arm. This includes the following two sub-modules: 1) the detection of different materials based on RGB data and their 3D localization inside a convex hull polygon based on depth data (*materials localization sub-module*) and 2) the localization of the highest (i.e. peak) point inside each material with respect to the base frame of the robot based also on depth data *(peaks localization sub-module)*. For achieving these tasks, RGB-D sensing has been used with a camera (ASUS Xtion PRO) installed in the scene, facing the materials to be manipulated. The RGB and depth (point cloud) data are given in an organized 640×480 grid-based structure at $30Hz$, which allows fast nearest neighborhood computations.

Materials Localization. For the material identification, we apply a color-based region growing segmentation. In particular, given a set of colored point cloud data acquired from the visual sensor, we first transform the cloud in the robot's base frame, i.e. the z-axis upwards and the y-axis facing left. Then, we apply two filters; first a pass-through filter to remove points below the robot base and second a crop-box filtering for keeping points only in the working space of the robot. An important aspect during this process is to keep the point cloud in its original grid organization structure, and thus instead of removing points setting them to NaNs. For the remaining non-filtered points, we apply a region growing method to classify points of similar color. The algorithm works in two stages. In the first stage, the points are sorted according to their curvature and are selected sequentially as seeds starting from the one with the minimum curvature value. This is done since growth from a point with minimum curvature, i.e. the flattest area, reduces the number

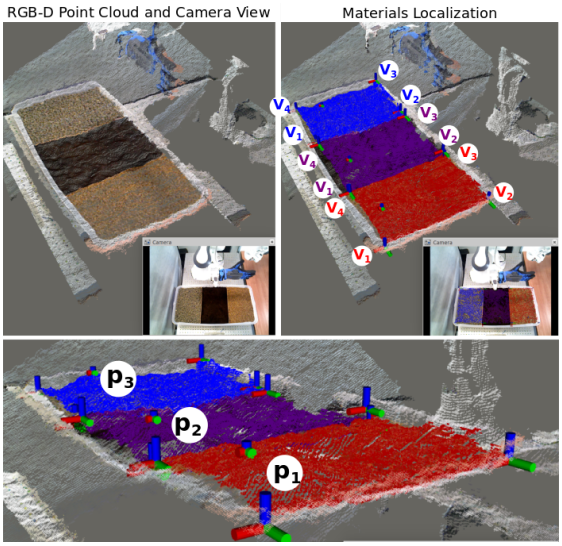

Fig. 3: The visual perception module.

of segments. If the neighboring points of a seed have similar color values they are classified as same material, until there are no other neighbors. This process continues until no other seed points remain. In the second stage, clusters whose color is close to the neighbor cluster or their size is small, are merged. The result of this process is visualized in Fig. 3. For each point cloud cluster we generate its convex hull polygon on the xy-plane. This polygon can be simplified to the 2D bounding box of the extreme points in each material's x and y directions. The vertices of each polygon (i.e., V_1, V_2, V_3, V_4 for the rectangle) are stored and passed to the FSM, as visualized in Fig. 3.

Peaks Localization. Having localized the different materials, it is straightforward to localize their peak point p_i, for material i, as this with the maximum z-value, while the center of the material can be defined as the average of the encapsulating polygon's vertices.

E. Finite State Machine

To enhance system autonomy by managing the transitions between the states (see Fig. 2), we introduced a Finite State Machine. This unit receives the data coming from the *Visual perception module*, elaborates them to define the target poses for the *Trajectory planner*, and defines if an interaction with the environment is expected to activate the *Self-tuning impedance unit*.

The FSM is mainly divided into four states. The first one, "Workspace definition", receives from the "Materials localization" unit the vertices of the polygons describing the materials in the robot workspace, and stores them in a data structure. In this way, the system acquires the knowledge about the areas where the robot will be moved in the next phases. The "Exploration" state is designed to identify the k_{st} parameter for every material. The robot, while holding a stick-like tool at the end-effector, in a compliant way reaches the leftmost material, described by the material polygon vertices given in the previous step. After dunking into the substance, the interaction expectancy value of the *Self-tuning impedance unit* is activated, since a contact with

the environment is predicted, and the robot follows a point-to-point motion described inside the material area adapting the impedance parameters as illustrated in Sec. II-B. At the end of this movement, the resulting $\boldsymbol{k}_{st}$ is stored in a data structure. Then, the robot pulls out the tool and repeats this phase for every material, until the impedance parameter of the rightmost one has been identified.

During the third state, "Materials distribution", the perception module comes again into play. The highest point of each substance is detected by the "Peaks localization" unit, sent to the FSM and stored in the data structure. These Euclidean points are used as the starting points of the trajectories designed in the next and final state, i.e. "Task", to make sure that some material will be found in that area. In this state, the robot holds a scooping tool, like a small shovel or a scoop. Starting with the default compliance $\boldsymbol{k}_{min}$ set in all the Cartesian axes, the robot interacts with the materials based on the specific task it has to carry out. For every material, the robot reaches its highest point and scoops some of it through a scooping motion that goes from the peak point towards the center of the material polygon. Next, it moves where it needs to put the material, it pours it in a container and it starts over with the next scheduled material, based on the task sequence. This time, when the interaction expectancy value is activated, the starting $\boldsymbol{k}_{st}$ is set differently for every material as the value that the framework learned in the "Exploration" state and stored in the data structure. In this way, during the "Task" state, the robot does not lag behind from the beginning and can carry out the task in a precise manner. In certain situations, it is possible that the material can change its viscous properties over time, either because of external circumstances or due to the intrinsic properties of the substance. For this reason, the impedance parameters can be subject to changes also in the "Task" state.

III. Experimental Setup

The software architecture of the robot relies upon the robotics middleware Robot Operating System (ROS) using C++ as client library. Every module described in Sec. II is implemented as a ROS node and data among units are exchanged via ROS messages on the ROS topics depicted in Fig. 2 through the publisher/subscriber design pattern.

We conducted experiments using two Franka Emika Panda robotic arms: one to accomplish all the phases of the proposed method and one to provide the container where the materials need to be poured. The proposed architecture was integrated in a customized version of *franka_ros* package, the ROS integration for Franka Emika research robots. This package integrates *libfranka*, Franka Emika's open source C++ interface, into ROS Control. The communication between this interface and the robot is made possible thanks to the Franka Control Interface (FCI), that provides the current robot status and enables its direct control with an external workstation PC connected via Ethernet in real-time at a communication rate of 1 kHz.

The robot was equipped with an underactuated hand, i.e., the Pisa/IIT SoftHand [19]. An ASUS Xtion Pro RGB-D sensor was attached on a pole in front of the robot arm and calibrated with respect to the robot base frame.

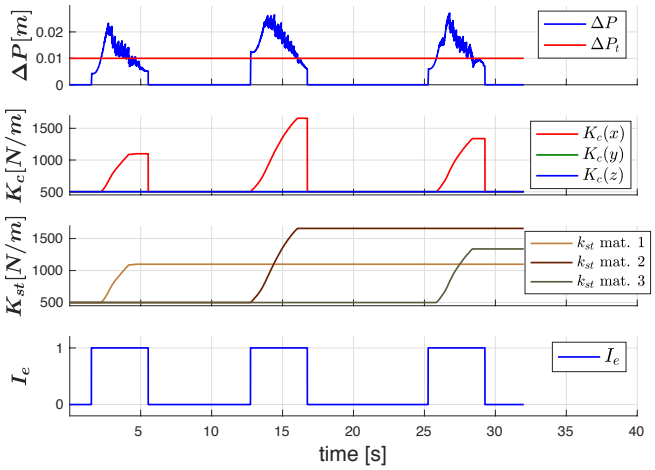

Fig. 4: "Exploration" state: when the interaction expectancy (I_e) Boolean value is True and the Cartesian error along the motion vector (ΔP) is above the threshold of $0.01m$, the self-tuning stiffness k_{st} increases and consequently does the Cartesian stiffness K_c in the direction of movement.

IV. Experiments

To validate the proposed method we carried out experiments in an agricultural robotics setup. Three different materials were placed in a container between the robot and the camera pole: soil, seeds and expanded clay. The choice of these materials was motivated both by their extensive use in agriculture and by the great difference among their viscoelastic properties. This allowed us to show different behaviors in the impedance parameters self-tuning and therefore to point out the validity of the method.

To illustrate the experiments, we follow the FSM flow. In the "Workspace definition" state, the FSM receives the 12 Euclidean points delimiting the areas of the three materials from the "Materials localization" unit of the perception module. Next, in the "Exploration" state, the robot holds a metal stick $27cm$ long. Based on the method explained at the end of Sec. II.B, $\boldsymbol{\alpha}$ was set to 20000, while $\boldsymbol{k}_{min}$ was set to $500N/m$, a value that guarantees a good compliance in case of unexpected collisions. In this state the robotic arm reaches the leftmost material, composed by seeds (mat. 1), and dunks the metal stick into it preparing for the exploration. Since an interaction is expected, at $t = 1.5s$ the interaction expectancy value is triggered, as shown in the fourth plot of Fig. 4, and the *Self-tuning impedance unit* is activated. The robot then moves along the x axis for $18cm$ with the tool dunked inside the material. As depicted in the first plot of Fig. 4, $\boldsymbol{\Delta P}$ goes beyond the threshold $\boldsymbol{\Delta P}_t$ set to $1cm$, and therefore $\boldsymbol{k}_{st}$ increases following (7). Consequently, the Cartesian stiffness along the direction of the movement, that in this case it is given entirely by x, is adapted. In this way, the Cartesian error along the motion vector, i.e. $\boldsymbol{\Delta P}$, decreases below the threshold and the value reached by $\boldsymbol{k}_{st}$ ($1100N/m$) is stored in the data structure. Next, the robot pulls out the

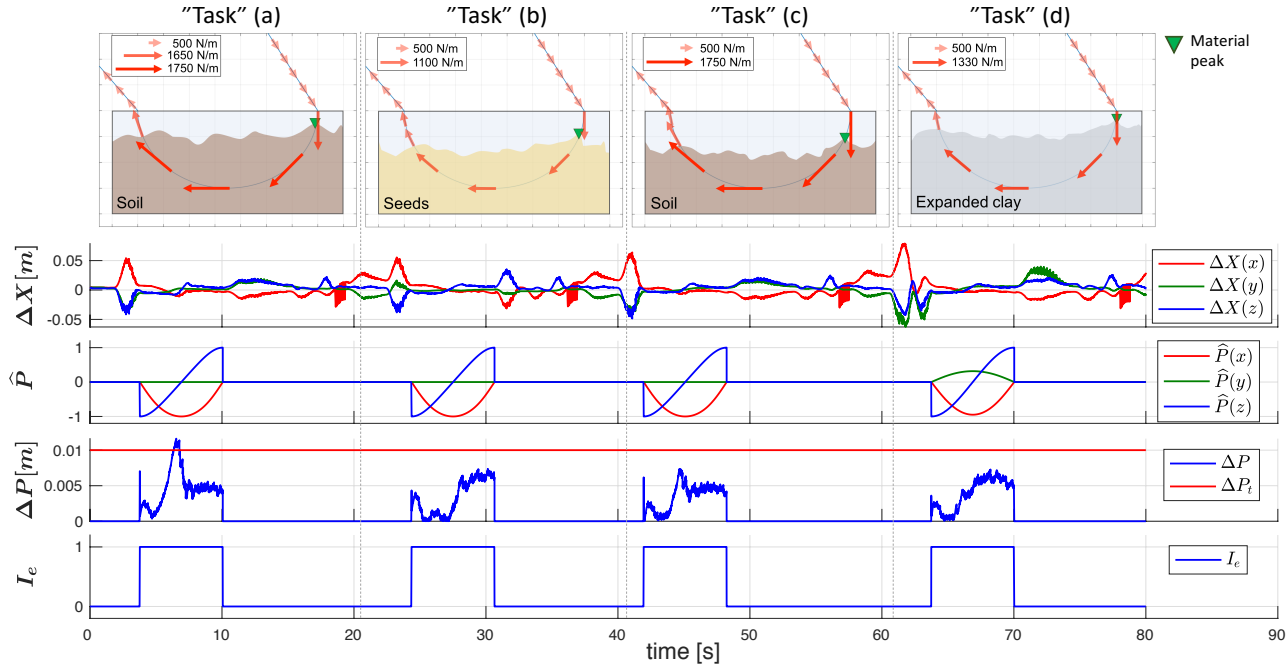

Fig. 5: The "Task" state subdivided into four substates: soil (a), seeds (b), soil (c) and expanded clay (d). In the upper figures, the red arrows represent the stiffness value $\boldsymbol{k}_{st}$ along the motion vector. High stiffness values are represented by longer and more vivid arrows, low stiffness values by shorter and faint arrows. The green triangles symbolize the material peak points detected by the *Visual perception module*. In the plots, ΔP reaches the threshold value only in the material where the viscoelastic properties changed from "Exploration" state, some water was added to the soil between the two states.

tool from the first material, and completes the exploration of the material in the center and the rightmost one in the same manner. The $\boldsymbol{k}_{st}$ values of the soil (mat. 2) and the expanded clay (mat. 3) are tuned, respectively, to 1650 and $1330 N/m$ as shown during $t = 10 - 30s$ in Fig. 4.

Once this learning step is completed, the robot goes back to its initial configuration, and the "Material distribution" state takes place. The perception module "Peaks localization" unit detects the highest point for every material, that are sent to the FSM, and stored in the data structure.

During the "Task" state, the robot hand grasps a scoop. This state is subdivided into four substates: the robot scoops, transports and pours in a plant plot, provided by another robot, some soil (a), some plant seeds (b), some other soil (c) and in the end some expanded clay (d). The four substates plots are illustrated in Fig. 5. In the upper figures, it is highlighted how the stiffness value $\boldsymbol{k}_{st}$, depicted by means of red arrows, changes along the motion vector $\widehat{\boldsymbol{P}}$ inside the interaction expectancy area, represented by the three material containers. Longer and more vivid arrows stand for higher stiffness values, while shorter and faint arrows stand for lower stiffness values.

Outside the containers, $\boldsymbol{k}_{st}$ is always set to $\boldsymbol{k}_{min}$, i.e. $500 N/m$. As soon as the tool enters the area where an interaction is expected, $\boldsymbol{k}_{st}$ assumes the value corresponding to the material, as shown by the arrows length and intensity change. If $\boldsymbol{\Delta P}$ remains below the threshold, $\boldsymbol{k}_{st}$ does not vary its value until no forces are detected, i.e. when the material scooping is over although the scoop is still inside the container. When this happens, $\boldsymbol{k}_{st}$ is reduced according to (8), as depicted in the last part of the trajectories performed inside the container; to avoid unnecessary changes due to negligible variations, we designed a moving average window to calculate $\boldsymbol{\Delta F}_{ext}$. Lastly, the robot exits the interaction expectancy area, and $\boldsymbol{k}_{st}$ is restored to $\boldsymbol{k}_{min}$.

Between the "Exploration" and the "Task" state, we put some water in the soil to change its viscoelastic properties. In "Task" (a), when the scoop enters the soil, $\boldsymbol{\Delta P}$ goes beyond the threshold ($t = 6.3s$), so the *Self-tuning impedance unit* is activated again. The soil $\boldsymbol{k}_{st}$ value passes from $1650 N/m$ to $1750 N/m$, as pointed out by the slight difference between the first and the other arrows inside the leftmost container.

The green triangles represent the highest point of each substance provided by the "Peaks localization" perception module. Note that, the axes of these figures are oriented to analyze the task from a lateral view to foster a deeper understanding. The direction of the motion is specified in the plot related to the three components of the normalized motion vector $\widehat{\boldsymbol{P}}$.

V. Conclusion and Discussion

In this work, we proposed a novel framework to enhance robot adaptability in unstructured and unknown environments. The framework was based on a self-tuning impedance controller, able to regulate the impedance parameters only on the direction of the motion vector, and activated just when an interaction with the external environment was predicted. It additionally included a visual perception module that improved the situation-awareness of the robot, localizing the surrounding materials and their peak points. We experimentally validated the presented framework in an agricultural task, demonstrating its high potential in online adaptation to interaction requirements of various materials.

REFERENCES

[1] D. Katz, A. Venkatraman, M. Kazemi, J. A. Bagnell, and A. Stentz, "Perceiving, learning, and exploiting object affordances for autonomous pile manipulation," *Aut. Robots*, vol. 37, no. 4, pp. 369–382, 2014.

[2] D. Katz, Y. Pyuro, and O. Brock, "Learning to manipulate articulated objects in unstructured environments using a grounded relational representation," *Robotics: Science and Systems IV*, p. 254, 2009.

[3] K. Kronander and A. Billard, "Learning compliant manipulation through kinesthetic and tactile human-robot interaction," *IEEE transactions on haptics*, vol. 7, no. 3, pp. 367–380, 2014.

[4] A. Dietrich, K. Bussmann, F. Petit, P. Kotyczka, C. Ott, B. Lohmann, and A. Albu-Schäffer, "Whole-body impedance control of wheeled mobile manipulators," *Aut. Robots*, vol. 40, no. 3, pp. 505–517, 2016.

[5] J. Lee, A. Ajoudani, E. M. Hoffman, A. Rocchi, A. Settimi, M. Ferrati, A. Bicchi, N. G. Tsagarakis, and D. G. Caldwell, "Upper-body impedance control with variable stiffness for a door opening task," in *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*. IEEE, 2014, pp. 713–719.

[6] A. Ajoudani, J. Lee, A. Rocchi, M. Ferrati, E. M. Hoffman, A. Settimi, D. G. Caldwell, A. Bicchi, and N. G. Tsagarakis, "A manipulation framework for compliant humanoid coman: Application to a valve turning task," in *14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, 2014, pp. 664–670.

[7] L. Righetti, M. Kalakrishnan, P. Pastor, J. Binney, J. Kelly, R. C. Voorhies, G. S. Sukhatme, and S. Schaal, "An autonomous manipulation system based on force control and optimization," *Aut. Robots*, vol. 36, no. 1-2, pp. 11–30, 2014.

[8] R. J. Anderson and M. W. Spong, "Hybrid impedance control of robotic manipulators," *IEEE Journal on Robotics and Automation*, vol. 4, no. 5, pp. 549–556, 1988.

[9] C. Schindlbeck and S. Haddadin, "Unified passivity-based cartesian force/impedance control for rigid and flexible joint robots via task-energy tanks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 440–447.

[10] C. Yang, G. Ganesh, S. Haddadin, S. Parusel, A. Albu-Schaeffer, and E. Burdet, "Human-like adaptation of force and impedance in stable and unstable interactions," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 918–930, 2011.

[11] F. Ferraguti, C. Secchi, and C. Fantuzzi, "A tank-based approach to impedance control with variable stiffness," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013, pp. 4948–4953.

[12] G. Xu, A. Song, and H. Li, "Adaptive impedance control for upper-limb rehabilitation robot using evolutionary dynamic recurrent fuzzy neural network," *Journal of Intelligent & Robotic Systems*, vol. 62, no. 3-4, pp. 501–525, 2011.

[13] E. Gribovskaya, A. Kheddar, and A. Billard, "Motion learning and adaptive impedance for robot control during physical interaction with humans," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4326–4332.

[14] Y. Li, S. Sam Ge, and C. Yang, "Learning impedance control for physical robot–environment interaction," *International Journal of Control*, vol. 85, no. 2, pp. 182–193, 2012.

[15] W. He and Y. Dong, "Adaptive fuzzy neural network control for a constrained robot using impedance learning," *IEEE Trans. on neural networks and learning systems*, vol. 29, no. 4, pp. 1174–1186, 2018.

[16] B. Nemec, N. Likar, A. Gams, and A. Ude, "Human robot cooperation with compliance adaptation along the motion trajectory," *Autonomous robots*, vol. 42, no. 5, pp. 1023–1035, 2018.

[17] P. Balatti, D. Kanoulas, G. F. Rigano, L. Muratore, N. G. Tsagarakis, and A. Ajoudani, "A self-tuning impedance controller for autonomous robotic manipulation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5885–5891.

[18] M. Kawato, "Internal models for motor control and trajectory planning," *Current opinion in neurobiology*, vol. 9, no. 6, pp. 718–727, 1999.

[19] A. Ajoudani, S. B. Godfrey, M. Bianchi, M. G. Catalano, G. Grioli, N. Tsagarakis, and A. Bicchi, "Exploring teleimpedance and tactile feedback for intuitive control of the pisa/iit softhand," *IEEE transactions on haptics*, vol. 7, no. 2, pp. 203–215, 2014.

[20] A. Ajoudani, N. G. Tsagarakis, and A. Bicchi, "Choosing poses for force and stiffness control," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1483–1490, 2017.

[21] A. Albu-Schaffer, C. Ott, U. Frese, and G. Hirzinger, "Cartesian impedance control of redundant robots: recent results with the dlr-light-weight-arms," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2003, pp. 3704–3709.