An Online Flexible and Scalable Architecture for Human-Robot Cooperation

Kourosh Darvish, Barbara Bruno, Enrico Simetti, Fulvio Mastrogiovanni, Giuseppe Casalino

Abstract—Modern manufacturing paradigms introduce the need for collaborative robots, able to naturally cooperate with humans in a partially-structured and dynamic environment.

In this article, we extend FlexHRC, an architecture for flexible and collaborative manufacturing robots, with a flexible and scalable hierarchical software architecture which enables the autonomous cooperation of the robot with the human in partially-structured and dynamic environments. To do so, the architecture encompasses three interleaved perception, action, and representation levels. We have tested the proposed system with a dual-arm manipulator cooperating with a person to assemble a table with one tabletop and four legs.

I. INTRODUCTION

Consumer-driven markets have introduced a new paradigm to production, called social manufacturing, according to which consumers should be fully involved in the production process. As a consequence, a flexible and agile production process is envisaged, and it is expected that frequent reconfigurations at the shop-floor and warehouse levels will be necessary to cope with diversified demands on tight schedules [1]. Human-Robot Cooperation (HRC) is considered a very promising approach to deal with highly dynamic manufacturing processes, because it combines the flexibility of human operators with the repeatability and precision of robots. The robot design and autonomy is the key element for the flexibility, whereas the human ability to decompose complex scenarios into simpler ones and the hierarchical representation of a cooperation scenario by the robot enables the necessary scalability. The approach, however, also entails a number of challenges, including physical safety, psychological assessment of operators’ well-being, human-centric design, natural and intuitive information exchange and communication, as well as robot autonomy and learning [2], [3].

A natural and efficient interaction assumes different degrees of autonomy in the robot’s decision making processes and human-like communication capabilities. On the one hand, if the collaborative robot is to perform simple tasks, it should execute them without the need for human intervention or directives. On the other hand, the robot should recognize at runtime and adapt to human operators’ intentions, balancing the need to meet common goals and to limit the cognitive burden on the person [4].

In this paper, we present a novel approach to human-robot cooperation in shop-floor settings aimed at increasing the online flexibility, scalability, the adaptation capabilities, and the controlled autonomy of collaborative robots, which is integrated in the FlexHRC architecture introduced in [4]. FlexHRC estimates future robot behaviors online, exploits such estimates to take decisions as far as cooperation is concerned, and allocates tasks to either humans or robots, reactively adapting to human operator decisions.

II. THE EXTENDED FLEXHRC ARCHITECTURE

A. Rationale

Figure 1 shows the three levels of the FlexHRC architecture, namely perception (in blue in the Figure), representation (in green in the Figure) and action (in red in the Figure). The perception level is composed of three modules, namely Object & Scene Perception, Knowledge Base and Human Action Recognition (HAR). The former two provide information about the state of the robot and the workspace and the latter about human actions. Task Representation and Planner form the representation level, where the cooperation process is defined and handled. The action level is responsible for managing the robot and consists of three modules, i.e., Robot Execution Manager, Controller and Simulator, the latter two being related to actions defined by the former and to be performed by the real or the simulated robot, respectively.

B. Perception level

The Object & Scene Perception receives depth information related to the robot’s workspace from an RGB-D sensor mounted on top of the robot, it applies a Euclidean metric for clustering the point cloud, and it uses the Random Sample Consensus (RANSAC) method to classify objects and tag those corresponding to primitive shapes with a semantic label [5]. Additionally, Principal Component Analysis (PCA) computes complementary features to determine the grasping poses of objects. These information are sent to and stored in the Knowledge Base, which can be considered as a data structure that maintains up-to-date the information related to the workspace (i.e., the objects therein) and the robot status.

The Human Action Recognition module works in two stages. Offline, it models human actions in terms of gestures using Gaussian Mixture Modeling and Gaussian Mixture Regression over inertial data provided by a wrist-worn inertial sensor [6]. Online, it computes the Malahanobis distance between the continuous data stream and the represented gesture models to classify operator motions, and therefore
actions, continuously providing the possibility of each modelled gesture to match the data stream. Finally, it analyses the possibility patterns of all gestures to recognize the execution of an action when performed by an operator [4].

C. Action level

The Planner can send to the Robot Execution Manager two types of commands: (i) requests for the execution of an action; (ii) requests for the simulation of an action.

In both cases, the Robot Execution Manager is given a high-level command and it is responsible for turning it into one or a sequence of commands that the robot Controller or Simulator can execute. The Robot Execution Manager reports back to the Planner about the success/failure of any executed/simulated action, on the basis of the feedback provided by the Controller and Simulator modules.

The Controller employs a Task Priority framework to control robot motions at kinematic level [7], while the Simulator module predicts the robot's behavior in its workspace by simulating the robot's closed loop kinematic model. We assume that the Controller can compensate the disturbances affecting the real robot's motion which are not simulated.

D. Representation level

The Task Representation module describes the cooperation task as an organized set of states and state transitions and online provides feasible states and state transitions to the Planner, which updates the Task Representation whenever there is a change in the status of the cooperation. The change can be caused by the human operator, and thus signaled to the Planner by the Human Action Recognition module, by external agents independently affecting the workspace, signaled by the Knowledge Base, or by the robot, and thus signaled by the Robot Execution Manager module, as we will see in the next Section.

To semantically formalize the cooperation process between human operators and robots, the Task Representation module employs AND/OR graphs. An AND/OR graph \( G(N,H) \) consists of a set of nodes \( N \) and a set of hyper-arcs \( H \). A node \( n \in N \) represents a state of the cooperation, whereas a hyper-arc \( h \in H \) represents a specific state transition among states. In particular, a hyper-arc \( h \) connects a set of child nodes \( N_C \subseteq N \) to a parent node \( n_P \in N \). The relation between child nodes in a hyper-arc is the logic AND, while the relation between different hyper-arcs inducing on the same parent node is the logic OR. Each hyper-arc \( h \in H \) corresponds to a sequence of ordered actions, \( A(h) \), to be executed by the human operator or the robot to reach the cooperation status described by the parent node. If all the actions associated with an hyper-arc or node are done in the right order, the hyper-arc or node is marked as solved. A node \( n_i \) is feasible if there exists a solved hyper-arc \( h_j \in H \) for which \( \text{parent}(h_j) = n_i \) holds. A hyper-arc \( h_i \in H \) is feasible if all its child nodes are solved. An AND/OR graph is solved if the root node is solved. To support the scalability of the architecture in the representation level, we extend the AND/OR graph to a hierarchical AND/OR graph representation, where a hyper-arc at a higher level embraces another AND/OR graph at a lower level.

The Planner module workflow is organized in two phases, the first offline and the second online. The offline phase loads all action definitions, their parameters, the semantic knowledge associated with the capability of agents to perform different actions, and, from the Task Representation module, the action sequences \( A(h_i) \) associated with each hyper-arc \( h_i \) in the loaded cooperation graph. During the online phase, as the human-robot cooperation process unfolds, the sequences of actions corresponding to currently feasible states and state transitions are evaluated. More precisely, the Planner receives the set of feasible states (i.e., graph nodes) and state transitions (i.e., hyper-arcs) from the Task Representation module, it finds the sequence of actions for human operators and robots that minimizes the cooperation cost, it assigns actions to them and, upon receiving perceptual information, it reports back to the Task Representation module about solved states and states transitions. During the online phase the Planner is responsible of the allocation of actions to either human operators or robots, as well as the definition of the most appropriate instances of action parameters. It fetches the current values of relevant parameters from the Knowledge Base and requests the simulation of the action to the Robot Execution Manager. Once all the possible robot allocations are simulated, the Planner compares them (also against the a-priori defined best combination involving a human operator)
according to a predefined utility function $J_i$ and determines the best combination of parameters and agents instances.

III. EXPERIMENTAL EVALUATION

To test the FlexHRC architecture, we consider the cooperative assembly of a table consisting of one tabletop and four legs. The cooperation involves three agents: human, robot left arm, and robot right arm. Each robot arm can perform the actions approach, transport, grasp, ungrasp, screw and unscrew. The action transport can be performed singly or jointly by the two robot arms. The human can perform the actions pick up (corresponding to the sequence approach-grasp-transport of the robot), screw, and put down (corresponding to the sequence transport-ungrasp of the robot). The human obtains information about the cooperation in natural language from the robot display. An action is defined as failed, both in reality and in simulation, if it is not successful within 30 seconds.

Figure 2 illustrates one example of human-robot cooperation for the table assembly task.

Figure 2 (1) shows the initial configuration of the workspace: the Planner initializes the cooperation state, receives the first feasible state transitions from the Task Representation module and simulates them to optimally allocate them to the agents. As a consequence (see scenes (2-8)) the robot approaches, grasps (with both arms) the tabletop and moves it to its final position. The tabletop weighs 1.4 kg, and this fact is purposefully not considered in the simulation. As expected, the controller is able to compensate the disturbance on the robot while performing the joint transportation action.

Once the tabletop is in its final position, the state of the cooperation is updated and the Planner assigns the placement of the legs to the agents. As a consequence (see scenes (9-10)) the robot approaches, grasps (with both arms) the tabletop and moves it to its final position. The tabletop weighs 1.4 kg, and this fact is purposefully not considered in the simulation. As expected, the controller is able to compensate the disturbance on the robot while performing the joint transportation action. Whenever the human interrupts a robot action, the robot is immediately commanded to go to its initial pose.

At scene (11) one leg and the tabletop are connected and the system, upon analysing the workspace, again updates its Task Representation. The Simulator generates and tests eight simulation possibilities. On the basis of the simulation results, the Planner assigns the action to the robot left arm, but, as the robot begins moving, the human decides to perform an action and connect one of the legs to the tabletop (scenes (9-10)). This situation tests the planning flexibility. Whenever the human interrupts a robot action, the robot is immediately commanded to go to its initial pose.

In this article we propose and examine methods for dynamic task planning, simulation and allocation for cooperative robots. The methods, which extend the previously proposed FlexHRC architecture, allow both human operators and robots to deviate from the optimal plan, ensuring that all other agents adapt their actions accordingly. These features are tested in a real cooperative scenario, in which a dual-arm manipulator cooperates with a human operator to assemble a small table composed of a tabletop and four legs. The experiment proves that the proposed architecture is able to dynamically simulate actions, allocate tasks and recover from unexpected events by re-planning.

REFERENCES

Fig. 2: One example of human-robot cooperation for the table assembly task.