Towards Reactive Robot Applications in Dynamic Environments

Anders Lager[1,2], Giacomo Spampinato[1], Alessandro V. Papadopoulos[2], Thomas Nolte[2]
[1]ABB AB, Västerås, Sweden
[2]Mälardalen University, Västerås, Sweden

***Abstract*—Traditionally, industrial robots have been deployed in fairly static environments, to perform highly dedicated tasks. These robots perform with very high precision and throughput. However, nowadays there is an increasing demand for utilizing robots in more dynamic environments, also performing more flexible and less specialized operations — high mix/low volume. Both traditional industrial robots and force-limited robots are used in collaborative, dynamic environments. Such robot applications introduce new challenges when it comes to efficiency and robustness. In this paper, we propose an architecture for reactive multi-robot applications in the context of dynamic environments, and we analyze the main research challenges that must be tackled for its realization. A logistics use case, with robots picking customer orders from the shelves of a warehouse, is used as a running example to support the description of the key challenges.**

I. Introduction

Industrial robots are commonly used in fairly static environments where they perform specialized tasks with excellent precision and high throughput. However, deployment of industrial robots continues to expand to new scenarios and applications, and there is an increasing demand for utilizing robots in more dynamic environments, also performing more flexible and less specialized operations [1]. Industry is undergoing a transition from traditional caged robots capable of handling all payloads, fast and precise, to newer collaborative robots that can safely work alongside humans, and that can be fully integrated into workbenches [2]. The combination of a continuously expanding market for robotics, and new emerging robotics applications introduce new challenges when it comes to efficiency and robustness.

In such a context, collaborative robots are playing a significant role [1], thanks to the lower installation cost and their ease-of-use. Safety requirements are handled in new ways, e.g., by replacing fences with safety sensors. The safety aspects of collaborative robots have been addressed by recent standardization efforts [3]. The reduced footprint and increased flexibility of these robots have also changed the way robots are deployed by traditional customers categories, e.g., the automotive industry. For example, collaborative robots are performing assembly or inspection on car bodies side by side with human workers doing other tasks in parallel [4].

Improving the ability of industrial robots to cope with a more dynamic and less structured environment is therefore becoming the main challenge, especially for collaborative robots. In particular, the robots should be able to complete assigned tasks in a safe manner while also maintaining a high productivity and quality, when exposed to disturbances and unexpected events.

In this paper we analyze the main challenges encountered by industrial robot applications in dynamic environments.

A logistics use case inspired by the Amazon picking challenge[1] is presented and discussed in relation to the presented challenges. To overcome the problems with improved efficiency and robustness, we conclude that this type of system needs to be reactive by design. We propose an architecture for reactive multi-robot systems and we analyze the research challenges for its realization.

II. Problem statement

Industrial robots are typically deployed in a structured environment behind protective fences. The actions from other actors, e.g., other robots, that affect the environment are therefore considered to be predictable. As a consequence, the robot motion is actuated in a "blind" way, typically with no visual feedback. In some applications, additional specialized sensor data can be used to fine adjust the robot path in real-time, to compensate for predictable deviations of the application task, e.g., deviations of a work piece that shall be processed.

When the protective fences are removed, and the human operator is introduced within the workspace of the robot, such an approach falls short due to the dynamic and semi-structure environment where the robots operate. Humans can be monitored with safety certified sensors and reactive actions can be taken to maintain safety, e.g., by lowering the speed or stopping the robot [3]. However, reducing the speed or stopping the robot reduces productivity. New reactive strategies capable of maintaining a high level of productivity and safety at the same time, while operating in shared working space, are needed.

Robots need a degree of autonomous behavior where the detailed plan of motion and process operations are generated from a more general job description that is aided by perception of the environment. For example, high-level programming can be made on the skill level [5], that in turn is compiled to motion and process primitives from object and gripper

This work was funded by The Knowledge Foundation (KKS), project ARRAY, and by The Swedish Foundation for Strategic Research (SSF).

[1]https://amazonpickingchallenge.org/

characteristics and process strategies, e.g., derived from heuristics [6]. However, as pointed out in [7], a robot must be able to combine high level planning with reactiveness for robust handling of tasks in a dynamic environment.

A robot application in a *dynamic environment* will be more be exposed to unexpected problems, e.g.:

- Failures caused by inaccurate knowledge or perception of the environment.
- Failures caused by unexpected disturbances from humans or other actors in the environment.
- Inefficiency caused by unexpected variation of conditions. The current plan is still valid but has a significant potential for improvements.

To steer towards the goal of efficient reactive strategies, we plan to address the following research questions in this work:

Q1. How can the unstructured dynamic environment be modeled?

Q2. How to design robotics solutions able to react in a timely and efficient way to unforeseen events?

Q3. What are the strengths and limitations of existing reactive strategies?

III. Logistic Use Case

Warehouses, workshops and other facilities require efficient storage and retrieval of objects into/from storage locations. Some of these systems can be fully automated, while others still require human participation in the process. The use case of picking mixed types of objects from warehouse shelves with a robot arm into containers was addressed in the competition Amazon Picking Challenge (APC) [8]. It is performed in a semi-structured environment that requires autonomous robots endowed with perception, motion-, grasp- and task planning abilities. In the competition, the use case was somewhat simplified compared to a real scenario, e.g., with a limited number of object types that were placed side by side and lightly packed in the central area of a stationary storage unit. Some teams used a stationary robot arm while other teams used a mobile platform to carry the arm. After the competition, the teams were asked what they would change in their various approaches to solve the problem. 8 of 22 responding teams answered they wanted to include more reactive control, e.g., force feedback or visual servoing. 17 of 25 teams agreed (strongly or somewhat) to the statement that "perception needs to be better integrated with motion planning".

In a real picking scenario, the complexity is increased. The picking robots need to handle high-mix low-volume with an almost unlimited number of object types with variation of shapes, sizes, friction, deformability and other properties. Objects are packed more densely and their visibility may be limited or occluded. On top of this, other uncertainties, e.g., intervention from humans or other autonomous actors, will further increase the problems. To overcome these challenges, a robot system needs to be reactive by design.

Grasping an object is a challenging task. A grasping plan may fail due to unknown properties of the object, e.g., deformability and friction. Unknown properties of neighbouring objects or the gripper may also have an influence. In a grasping scenario, reactive strategies become critical capabilities to mitigate the consequences of unstable grasps and increase the efficiency [9]. Even when grasping an object succeeds with a stable grip, the movement of the object in the gripper while grasping can be hard to predict and compensate for. In [10] and in [11] methods for underactuated grippers based on constrained optimization and on machine learning are proposed to compensate for in-hand object motions while grasping. Unexpected movements of the object while grasping may cause the grasped object to knock down neighboring objects, e.g., while transferring the object after grasping.

While picking is ongoing, it is desirable that the picking robot is reactive to the following events:

- Unexpected movements of the object being manipulated. The object may start to slide in the gripper or in-hand motion may cause an unexpected grasping location. As a consequence, the object itself or neighboring objects may drop from the shelf.
- Unexpected movements of neighboring objects. Neighboring objects may become moved by influence of the manipulated object or by the gripper in an unexpected way.
- Unplanned speed changes of the mobile platform. Such speed changes may require immediate adjustments of the manipulation task to prevent a grasping failure or a collision with the robot arm.

IV. Architecture for Reactive Replanning

A *reactive strategy* consists of two system abilities: (i) *Reactiveness*, i.e., an ability to decide when reactive planning is advantageous to avoid failures or improve efficiency. Reactiveness requires an evaluation of the current plan that considers the motion of objects and actors in the environment. (ii) *Reactive replanning*, i.e., an ability to find and execute a new and more efficient plan, that will replace the current plan.

A new plan may constitute a smaller or a bigger adjustment of the current plan. The current plan may be adjusted on one or more levels:

- The plan for the ongoing robot trajectory.
- The plan for the ongoing robot task. A robot task can be seen as a limited action of the robot, e.g. pick up an object from a location. The constituents of a robot task are a sequence of path segments and process interactions that needs to be completed to finish the task. A process interaction is a planned interaction with process equipment, e.g. open the gripper. Normally, a process interaction needs to be coordinated with the robot trajectory, e.g., the gripper shall be opened at a specific place location.
- The scheduling of different robot tasks. Tasks can often be executed in different sequences or with allocation of different resources needed for their completion (e.g., a robot, a gripper, an inspection camera, a processing machine etc.). Multiple robots enable allocation of different robots or parallel execution of tasks. The rescheduling possibilities can depend on overall goals (e.g., fill a pallet

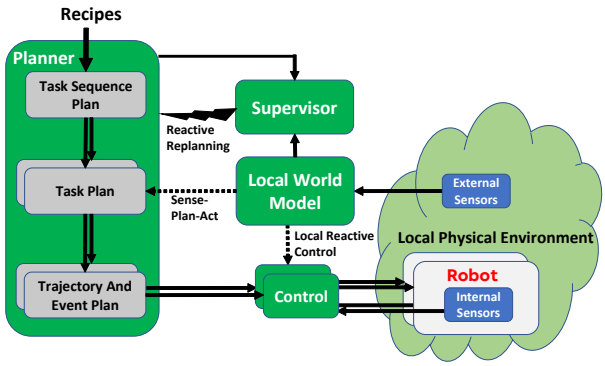

Fig. 1. Architecture for reactive multi-robot applications.

with parts), resource availability (e.g. need a specific processing machine) and dependencies between different robot tasks (e.g., after picking up a part, it needs to be placed).

To address Q1 and Q2, we propose an architecture for the implementation of a reactive strategy for industrial multi-robot applications that considers replanning on different levels. An overview of the proposed architecture is shown in Fig. 1. The light green cloud symbol represents the local physical environment with multiple robots (white) and sensors (blue). The dark green boxes represent software components for controlling the robots. The grey boxes represent different levels of the robot application plan. The arrows represent flows of information. The dashed arrows represent optional flows of information. The flash symbol represents an event that is fired under certain conditions.

Robots are deployed to perform tasks in a *Local Physical Environment*. An *External Sensors* component perceives areas of the *Local Physical Environment*, e.g., using cameras.

A *Local World Model* component keeps data on parts, e.g., to be processed, obstacles (fixed objects, restricted zones), and actors (humans, mobile robots) present in the *Local Physical Environment*. Data may include non-observable properties, e.g., weight and surface friction. Dynamic data are updated continuously from processing of incoming sensor data. Dynamic objects are detected, located and classified. The *Local World Model* can predict the near future movements of objects.

A *Planner* component plans the execution of incoming recipes. A *recipe* is a high level job description specifying a sequence of part processing needed to finalize a job. Examples of jobs: Create an assembly or palletize a pallet pattern. The *Planner* may handle multiple recipes in parallel. For each recipe, the *Planner* generates a *Task Sequence Plan*. The *Planner* also performs execution control of the task sequence plans. It has a scheduling function that continuously selects and assigns *Tasks* to be executed by the *Robots*. For each *Task*, an allocation is made of *Robots* and other needed resources, e.g., tools, parts, conveyors or processing machines. The *Local World Model* provides data on available resources and their spatial relationships. A *Task Plan* is generated for each *Robot*. If a *Task* requires more than one robot, e.g., assemble two parts using two *Robots*, a *Task Plan* is generated for each *Robot* and the parallel execution of the *Task Plans* is coordinated.

The *Task Plan* is a sequence of path segments and related process interactions. When a *Task Plan* is generated, the *World Model* may provide data on location of parts, obstacles and actors. Optionally, if some of these data are continuously updated by the *External Sensors*, e.g., arrived parts to be processed, this can be regarded as a sense-plan-act step [12]. At the fine-grained level, the *Planner* generates a *Trajectory And Event Plan* for each *Robot*, consisting of trajectory references to be executed by a *Control* component. It also generates process signal events to be executed synchronized with the trajectory.

A *Control* component receives trajectory references and controls the motions of a *Robot*. It receives feedback from *Internal sensors* of the *Robot*, e.g., axis resolvers. Optionally, it can receive dynamic location data of nearby objects from the *World Model*. If the *Control* uses this data as control input e.g. for collision avoidance, this can be regarded as local reactive control [12].

A *Supervisor* component compares the current plan with predictions of the *World Model*. The supervisor decides when it is advantageous to perform a reactive planning in an efficiency perspective. Whenever a conflict or opportunity (e.g. a failure or inefficiency) is detected that may invalidate the current plan, a *Reactive Replanning* event is generated. An important factor that is beneficial for early conflict/opportunity detection is the access of near future data for both the planned action of the Robot and the predicted movements of parts and actors in the *Local Physical Environment*.

When the *Planner* receives a *Reactive Replanning* event, it replaces the current plan with a new or modified plan. It needs to decide on what level the plan shall be changed, generate the new plan and prepare a reactive transition between the old plan and the new plan. This event shall not be mixed up with *reactive planning* [12] which is a strategy that incorporates both global planning and local reactive control but with continuous updates. *Reactive Replanning* is intended to fill a gap between Sense-Plan-Act strategies where the plan is setup and followed in a stepwise manner and Local Reactive Control where the plan is updated continuously on a trajectory level. *Reactive Replanning* follows the setup plan with continuous supervision and will only update the plan when needed. One advantage with this strategy is the ability to make reactive replanning on different levels, not only on a trajectory level. By being able to consider the robot's overall goal, the variability of replanning alternatives are increased which enables a potential for higher productivity.

V. Research Challenges

For the realization of the proposed architecture, requirements and research challenges are identified for the architectural elements.

External Sensors. Reactive robot applications require the sensors to be sampled fast with enough resolution and throughput. The communication interface must be able to handle the throughput with low latency. Vision sensors are often needed

to perceive movable parts and actors in the *Local Environment*, e.g., shelves and objects on the shelves. Tactile sensors may be used to measure contact surfaces and pressure between objects, e.g., the gripper and a grasped part.

Local World Model. The *Local World Model* must be able to give fast predictions of object movements. The processing of sensor data must be fast and able to handle the sensor throughput. It should be able to predict the near future motion of objects and actors with some confidence. Motion prediction may be based on extrapolation of the latest movement samples, e.g., [13], or mined patterns of movements [14]. Human intention detection/prediction can also potentially be used to improve the efficiency of robots having collaborative tasks with humans [15], [16]. For a picking application, detecting slipping of a grasped object due to an unstable grip is valuable. [9] presents a slipping detection method for a robotic finger. In general, hardware computing capacity and the efficiency of algorithms must be addressed.

Supervisor. The Supervisor must be able to predict the need for replanning fast, accurately and with enough throughput to consider recent changes in the Local Environment. One particular challenge is to derive criteria that will make it advantageous to perform a reactive replanning. Examples:

- Criteria for failure detection/prediction: The probability of failure for the current plan has reached a certain level.
- Criteria for opportunity detection: The efficiency of the current plan has been reduced to a certain level.

In general, hardware computing capacity and the efficiency of algorithms must be addressed.

Reactive Replanning. The computation of a new/updated plan needs to be fast [17]. One problem is to generate a new plan that considers ongoing and unfinished process interactions of the current plan. On the other hand, there is the problem of performing the transition from the current plan to the new plan, and to handle the needed planning time in this transition. Preferably, the transition is made on-the-fly, i.e., without stopping the robot movement.

Replanning can be done on three levels, and one problem is how to select the level of replanning: (i) *Trajectory replanning*, i.e., changing the path or the speed of the robot to avoid a collision with the robot gripper and an object. (ii) *Task replanning*, i.e. make a new plan for an ongoing task and possibly introducing new tasks to deal with what caused the replan. (iii) *Task rescheduling*, i.e., changing the scheduled execution sequence of tasks or changing task allocation among the different robots and resources. Additionally, it may include interruption and rescheduling of an ongoing task.

To enable replanning on the three levels, the *Planning* component must be able to manage the planning from the high-level job description to the fine grained planning. The planning should be structured in a form that supports flexible and reactive replanning on all three levels while still accomplishing the job description. In order to improve replanning efficiency, it should also be able to leverage on the variability of the task, process and motion sequence while maintaining constraints [18].

VI. Summary and work-in-progress

In this paper we have proposed an architecture to improve the efficiency and robustness of industrial robot applications in dynamic semi-structured environments. We have highlighted the main challenges associated with such a problem and the need for reactive approaches. Our ongoing work concerns the implementation and verification of different parts of the proposed architecture (Q1, Q2), and the comparison with alternative architectures and reactive approaches (Q3), e.g., [18]–[21].

References

[1] A. Ajoudani *et al.*, "Progress and prospects of the human–robot collaboration," *Autonomous Robots*, vol. 42, no. 5, pp. 957–975, 2018.

[2] G. Litzenberger, Int. Federation of Robotics (IFR). (2019) World robotics 2018 industrial and service robot presentation. [Online]. Available: https://ifr.org/

[3] Int. Organization for Standardization Standard ISO/TS 15066:2016, "Robots and robotic devices-collaborative robots," 2016.

[4] D. Greenfield. (2018) Collaborative robots in automotive manufacturing. [Online]. Available: https://www.automationworld.com/article/industry-type/discrete-manufacturing/collaborative-robots-automotive-manufacturing

[5] M. Haage *et al.*, "Declarative-knowledge-based reconfiguration of automation systems using a blackboard architecture," in *11th Scandinavian Conf. on Artificial Intelligence*, 2011, pp. 163–172.

[6] A. Pichler and C. Wögerer, "Towards robot systems for small batch manufacturing," in *ISAM 2011*, 2011, pp. 1–6.

[7] R. P. Bonasso *et al.*, "Experiences with an architecture for intelligent, reactive agents," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 237–256, 1997.

[8] N. Correll *et al.*, "Analysis and observations from the first amazon picking challenge," *IEEE Trans. on Automation Science and Engineering*, vol. 15, no. 1, pp. 172–188, 2018.

[9] R. A. Romeo *et al.*, "Slippage detection with piezoresistive tactile sensors," *Sensors*, vol. 17, no. 8, 2017.

[10] M. Liarokapis and A. M. Dollar, "Post-contact, in-hand object motion compensation for compliant and underactuated hands," in *RO-MAN 2016*, 2016, pp. 986–993.

[11] ——, "Learning the post-contact reconfiguration of the hand object system for adaptive grasping mechanisms," in *IROS 2017*, 2017, pp. 293–299.

[12] D. Kappler *et al.*, "Real-time perception meets reactive motion generation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1864–1871, 2018.

[13] Y. Tao *et al.*, "Prediction and indexing of moving objects with unknown motion patterns," in *SIGMOD/PODS 2004*, 2004, pp. 611–622.

[14] H. Jeung *et al.*, "A hybrid prediction model for moving objects," in *ICDE 2008*, 2008, pp. 70–79.

[15] Y. Shen *et al.*, "A design approach for incorporating task coordination for human-robot-coexistence within assembly systems," in *SysCon 2015*, 2015, pp. 426–431.

[16] A. V. Papadopoulos *et al.*, "Generation of human walking paths," *Autonomous Robots*, vol. 40, no. 1, pp. 59–75, 2016.

[17] B. Miloradović *et al.*, "Extended colored traveling salesperson for modeling multi-agent mission planning problems," in *ICORES 2019*, 2019, pp. 237–244.

[18] R. M. Zlot, "An auction-based approach to complex task allocation for multirobot teams," Ph.D. dissertation, CMU, 2006.

[19] N. J. Nilsson, "Teleo-reactive programs for agent control," *J. Artif. Int. Res.*, vol. 1, no. 1, pp. 139–158, 1994.

[20] N. M. Ceriani *et al.*, "Reactive task adaptation based on hierarchical constraints classification for safe industrial robots," *IEEE/ASME Trans. on Mechatronics*, vol. 20, no. 6, pp. 2935–2949, 2015.

[21] A. Hentout *et al.*, "Multi-agent architecture model for driving mobile manipulator robots," *Int. Journal of Advanced Robotic Systems*, vol. 5, no. 3, p. 30, 2008.