The 26th International Conference on Automated Planning and Scheduling

Proceedings of the 1st Workshop on

Planning, Scheduling and Dependability in Safe Human-Robot Interactions

Edited by:

Ali Shafti, Andrea Orlandini

London, UK, 13/06/2016

Organising Committee

Ali Shafti
King's College London, UK
Kaspar Althoefer
Queen Mary University of London, UK
Helge A. Wurdemann
University College London, UK
Amedeo Cestra
National Research Council of Italy (CNR-ISTC)
Andrea Orlandini
National Research Council of Italy (CNR-ISTC)
Iñaki Maurtua
IK4-Tekniker, Spain

Program Committee

Kaspar Althoefer (Queen Mary University of London, UK)
Saddek Bensalem (VERIMAG, France)
Stefano Borgo (CNR-ISTC, Italy)
Amedeo Cesta (CNR-ISTC, Italy)
Marcello Cirillo (SCANIA, Sweden)
Michael Fisher (University of Liverpool, UK)
Sami Haddadin (Leibniz Universitat Hannover, Germany)
Klaus Havelund (NASA-JPL, USA)
Joachim Hertzberg (University of Osnabrueck, Germany)
Petar Kormushev (Imperial College London, UK)
Daniele Magazzeni (King's College London, UK)
Fulvio Mastrogiovanni (Universita' Degli Studi Di Genova, Italy)
Iñaki Maurtua (IK4-Tekniker, Spain)
David Musliner (Smart Information Flow Technologies, USA)
Andrea Orlandini (National Research Council, CNR-ISTC, Italy)
Jose Saenz (Fraunhofer IFF, Germany)
Ali Shafti (King's College London, UK)
Helge A. Wurdemann (University College London, UK)
Jeremy Wyatt (University of Birmingham, UK)

Foreword

SafePlan aims to bring together experts active in the field of planning and scheduling (P&S) with those in human-robot interaction with particular emphasis on safety. The sector experiences a paradigm shift from the traditional heavy-duty robot operating separated from the human worker in a fenced area to robots that work close to the human, adapting to the movements of the human and possibly even interacting with them. In this regard, tools and methodologies for verification and validation (V&V) of P&S systems have received relatively little attention. Therefore, important goals of the workshop are also to focus on interactions between P&S and V&V communities as well as to identify innovative V&V tools and methodologies when applied to P&S in human-robot collaboration scenarios.

The workshop is promoted together with the H2020 EU project FourByThree (http://www.fourbythree.eu) which aims to create modular, efficient, intuitive and safe robots that can work in close collaboration with humans. In this regard, the workshop aims at exploring the potential applications of P&S and/or V&V to robots as above operating in a modern factory environment where human-robot interaction is used as a means to pave the path for accelerated manufacturing whilst reducing costs.

The workshop is organized aiming to foster synergies with both the Robotics Track (http://icaps16.icaps-conference.org/robotics-track.html) chaired by Nick Hawes and Andrea Orlandini. In this regard, the workshop will serve as a means for those more involved in P&S to familiarise themselves with the challenges in safety critical applications, such as, for instance, human-robot collaborative applications where their expertise are indeed needed and can be readily applied.

Topics of interest include, but are not limited to:

- How safe are today's robots to allow human-robot interaction in shared workspaces?
- What are the best technologies currently available to achieve safe robots?
- Human perspective (including trust towards and acceptance of robotic systems);
- How can planning and scheduling be applied to the safe human-robot interaction problem?
- What role do validity, verification and dependability play in safe human-robot interactions?
- P&S for long-term autonomy in human-robot collaborative scenarios;
- Integrated planning and execution in robotic architectures;
- Human-aware planning and execution in human-robot interaction, including safety;
- Failure detection and recovery in P&S systems;
- Formal methods for robot planning and control;
- V&V of P&S models, using technologies such as static analysis, theorem proving, and model checking;
- Consistency and completeness of P&S models;
- Runtime verification of plan executions;
- Generation of robust plan controllers;

Additionally, we are particularly looking for information on specific domains in which the co-presence of planning and scheduling capabilities should merge with the requirement of safety guarantee. The organizers of the workshop will provide initial cases from the manufacturing domain but are welcoming descriptions of other domains where similar needs are present to create a community of practice to address these similar problems under different scenarios.

What follows are papers accepted at the SafePlan workshop.

Table of Contents

Interacting with collaborative robots in industrial environments: A semantic approach

Iñaki Maurtua, Izaskun Fernandez, Johan Kildal, Loreto Susperregi, Alberto Tellaeche, Aitor Ibarguren
{inaki.maurtua, izaskun.fernandez, johan.kildal, loreto.susperregi, alberto.tellaeche, aitor.ibarguren}@tekniker.es
IK4-TEKNIKER
Iñaki Goenaga 5, 20600 Eibar
Basque Country, Spain

Abstract

This paper presents a semantic approach to support multimodal interactions between humans and industrial robots in real industrial scenarios. This is a generic approach and it can be applied in different industrial scenarios. We explain in detail how to apply it in a specific example scenario and how the semantic technologies help not only with accurate natural request interpretation but also their benefits in terms of system maintenance and scalability.

Introduction

In modern industrial robotics, the safe and flexible cooperation between robots and human operators can be a new way to achieve better productivity when performing complex activities. Introducing robots within real industrial settings makes the interaction between humans and robots gain further relevance. The problem of robots performing tasks in collaboration with humans poses three main challenges: robots must be able to perform tasks in complex, unstructured environments, and at the same time they must be able to interact naturally with the humans they are collaborating with, always guaranteeing the safety of the worker.

The current work is carried out in the context of the *H2020 FourByThree*[1] project, which aims at developping a new generation of modular industrial robotic solutions that are suitable for efficient task execution **in collaboration with humans** in a safe way and are easy to use and program by the factory worker. The project will allow system integrators and end-users to develop their own custom robot that best answers to their needs. To achieve it, the project will provide a set of hardware and software components, ranging from low level control to interaction modules. The results will be validated in 4 industrial settings: Investment Casting, Aeronautical sector, Machining and metallic part manufacturing, in which relevant applications will be implemented: assembly, deburring, riveting and machine tending in a collaborative context.

A requirement for natural Human-Robot Collaboration including interaction is to endow the robot with the capability to capture, process and understand accurately and robustly requests from a person. Thus, a primary goal for this research is to analyze the natural ways in which a person can interact and communicate with a robot.

[1]`http://fourbythree.eu/`

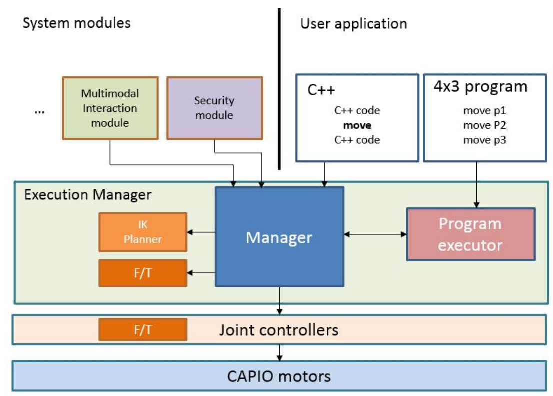


Figure 1: FourByThree project Architecture

Natural communication between humans and robots can happen through several channels, the main of which are voice and gestures. In this multimodal scenario, the information can be complementary between channels, but also redundant. However, redundancy can be beneficial (Bannat et al. 2009) in real industrial scenarios where noise and low lighting conditions are usual environmental challenges that make it difficult for voice and visual signals to be captured with clarity.

In this paper, we present a semantic approach that supports multimodal interaction between humans and industrial robots in real industrial settings that are being studied within the *FourByThree* European project. As mentioned earlier, the approach that we present is generic in the sense that it can be applied to different industrial scenarios by modifying the information about the environment in which communication takes place. After the approach description we introduce a case study corresponding to a real industrial scenario.

Related work

Over the last two decades, a considerable number of robotic systems have been developed showing Human-Robot Interaction (HRI) capabilities (Fong, Illah, and Dautenhahn 2003; Goodrich and Schultz 2007). Though recent robot platforms integrate advanced human-robot interfaces (incorporating body language, gestures, facial expressions, and speech) (R. Stiefelhagen and Waibel 2004; Burger, Ferrane, and Lerasle 2010) their capabilities to understand human speech semantically remains quite limited. To endow a robot with semantic understanding capabilities is a very challenging task. Previous experiences with tour-guide robots (Thrun et al. 1999; Gunhee et al. 2004) show the importance of improving human-robot interaction in order to ease the acceptance of robots by visitors. In Jinny's HRI system (Gunhee et al. 2004), voice input is converted to text strings, which are decomposed into several keyword patterns and a specialized algorithm finds the most probable response for that input. For example, two questions like 'Where is the toilet?' and 'Where can I find the toilet' are equally interpreted since the keyword pattern of 'where' and 'toilet' would be extracted from both cases.

Human-robot natural interactions have also been developed in industrial scenarios. For instance, in (Bannat et al. 2009) the interaction consisted of different input channels such as gaze, soft-buttons and voice. Although the latter constituted the main interaction channel in that use scenario, it was solved by command-word-based recognition.

SHRDLU is an early example of a system that was able to process instructions in natural-language and perform manipulations in a virtual environment (Winograd 1971). Researchers followed on that work towards extending SHRDLU's capabilities into real world environments. Those efforts branched out into tackling various sub-problems, including Natural Language Processing (NLP) and Robotics Systems. Notably, (MacMahon, Stankiewicz, and Kuipers 2006) and (Kollar et al. 2010) developed methods for following route instructions given through natural language. (Tenorth et al. 2010) developed robotic systems capable of inferring and acting upon implicit commands using knowledge databases. A similar knowledge representation was proposed by (Wang and Chen 2011) using semantic representation standards such as the W3C Web Ontology Language (OWL) for describing an indoor environment.

A generic and extensible architecture was described in (Rossi et al. 2013). The case study presented there included gesture and voice recognition, and the evaluation showed that interaction accuracy increased when combining both inputs (91%) instead of using them individually (56% in the case of gestures and 83% for voice). Furthermore, the average time for processing both channels was similar to the time needed for speech processing.

Our work is based on this extensible architecture, combining gesture and speech channels and adding semantic aspects to the processing.

Multimodal Interaction Semantic Approach

The approach proposed in this work aims at creating a human-robot collaborative environment in which interactions between both actors happen in a natural way (understanding by 'natural' the communication based on voice and gestures). We propose a semantic multimodal interpreter prototype that is able to process voice and gesture based natural requests from a person, and combining both inputs

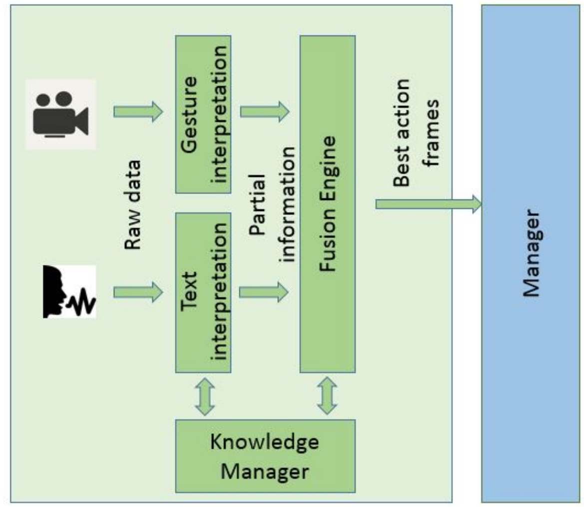


Figure 2: Multimodal Semantic Approach Architecture

to generate an understandable and reliable command for industrial robots. For such a semantic interpretation we have developed four main modules, as shown in Figure2.

A *Knowledge-Manager* module that describes and manages the environment and the actions that are feasible for robots in a certain environment, using semantic representation technologies. A *Text Order Interpretation* module that given a (request) text, it extracts the key elements on the text and translate them to a robot understandable representation, combining NLP and semantic technologies. A *Gesture Interpretation* module mainly for resolving pointing issues and some simple orders like stop an activity. And a *Fusion Engine* for combining the output of both text and gesture modules and construct a complete and reliable order for the robot.

These main modules are described in detail in the following subsections.

Knowledge Manager

The knowledge manager comprises ontologies that model environmental information of the robot itself including its own capabilities. In addition, the knowledge manager allows us to model the relationships between the concepts. These relationships are implicit rules that can be exploited by reasoners in order to infer new information from the ontology. As a result, reasoners can work as rule engines in which human knowledge can be represented as rules or relations.

Ontologies have many practical benefits. They are very reusable flexible at adapting to dynamic changes reuse, thus avoiding to have re-compile the application and its logic whenever a change is needed. Being in the cloud makes ontologies even more reusable, since different robots can exploit them, as was the case with e.g., RoboEarth (Di Marco et al. 2013).

Through ontologies, we model the industrial scenarios in which industrial robots collaborate with humans, in terms of robot behaviors, task/programs they can accomplish and the objects they can manipulate/handle from an interaction point of view. We distinguish two kinds of actions: actions that imply a status change on a robot operation, like *start* or *stop*, and actions related the robot capabilities such as *screw*, *carry*, *deburring* and so on.

Relations between all the concepts are also represented, which adds the ability for disambiguation during execution. This ability is very useful for text interpretation, since different actions can be asked from the robot using the same expression. For instance people can use the expression *remove* to request the robot to *remove this burr*, but also to *remove this screw*, depending on whether the desired action is *deburring* or *unscrew* respectively. If the relationships between the actions and the objects over which the action are performed are known, the text interpretation will be more accurate, since it will be possible to discern in each case to which of both options the expression *remove* corresponds. Without this kind of knowledge representation, this disambiguation problem is far more difficult to solve.

For task/programs we make an automatic semantic extension exploiting wordnet (Gonzalez-Agirre, Laparra, and Rigau 2012) each time the robot is initialized. In this way, we obtain different candidate terms referring to a certain task, which is useful for text interpretation mainly, as it is described bellow.

Text order interpretation

Given as input a human request in which a person indicates the desired action in natural language, the purpose of this module is to understand exactly what the person wants and if it is feasible to generate the necessary information for the robot. The module is divided into two main steps:

- The first step is based on superficial information, in the sense that it does not take into account the meaning of words in the context. Its only purpose is to extract the key elements from the given order.
- The second step attempts to identify the action that is asked for, considering the key elements in the given context.

For the first step, we apply natural language processing techniques using FreeLing, an open source suite of language analysis tools (Padró and Stanilovsky 2012). In particular, we apply a morphosyntactic and dependency parsing to a set of request examples from different people. In this way, we obtain the morphosyntactic information of every element and about the request itself. We revise the complete information manually and identify the most frequent morphosyntactic patterns. From them, we extract elements denoting actions, objects/destinations (target onward) and explicit expressions denoting gestures, such *there*, *that*, and so on. Following, we implement those patterns as rules, obtaining a set of rules that, given a FreeLing tagged sentence, is able to extract the key elements on it. Following, we implement those patterns as rules, obtaining a set of rules that are able to extract the key elements from the tagged sentence returned by FreeLing.

The aim of the second step is to identify which one of the tasks the robot is able to perform suits the request best, considering the key elements in it. We undertake this step by making use of the knowledge-base information described above. First, we verify if the identified actions are among the feasible tasks described in the knowledge base, and then we apply a disambiguation step using the target information, as explained before. This process results in the delivery of the best fits for the given input, from among the potential tasks obtained from the previous step.

The module output consists of frames, one for each potential task candidate, including information denoting gestures if it exists.

Figure 3: Pointing gesture mockup

Gesture Interpretation

Two kinds of gestures are addressed: pointing gestures and gestures for simple commands like stop/start. The case presented in this paper deals with pointing gestures that are recognized by means of point-cloud processing.

The initial setup consists of a collaborative robot and a sensor capable of providing dense point clouds, such as the ASUS Xtion sensor, the Microsoft Kinect sensor, or the industrially-available Ensenso system by IDS. The sensor is placed above the human operator and orientated towards the working area of the robot, so that the point cloud obtained resembles what the human operator in perceiving in the working environment (see Figure3).

The point cloud is then initially divided into two regions of interest (ROI), the first one corresponding to the gesture detection area, and the second one defining the working area of the robot where the pointing gesture will be applied.

With this setup, two main problems need to be solved for the interaction between the person and the robot to succeed:

1. Robust estimation of the direction of the pointing gesture.
2. Intersection of the pointing gesture with the working area of the robot.

Robust estimation of the pointing gesture The ROI for the pointing gesture detection is initially defined by specifying a cuboid in space with respect to the reference frame. In this case, the reference frame is the sensor frame, but it can also be defined using another working frame, provided a tf transformation exists between the frame used and the sensor frame. For robustness, the pointing gesture is defined using the forearm of the human operator. To identify the arm unequivocally, an euclidean cluster extraction is performed.

Intersection of the pointing gesture with the working area of the robot The main objective of a pointing gesture is to determine the point on the working area that is being pointed at. To identify this point, the points in the cloud corresponding to the pointing line are selected, from the furthest one all the way to the origin of the line that corresponds to the pointing arm. For each one of the points, a small cuboid is defined, and the ROI of the working area of the robot is filtered with it. If more than N points of the working area are present inside the small centered cuboid defined in the points of the projection line, an intersection has been found. The final intersection point that is published is the closest one to the origin of the projection line. As a threshold, a minimum euclidean distance value is defined in order to avoid detecting intersections corresponding to the proper point cloud of the arm that generates the pointing gesture.

Fusion Engine

The fusion engine aims to merge both the text and the gesture outputs in order to deliver the most accurate request to send to the executive manager. The engine consider different situations regarding the complementary and/or contradictory levels of both sources.

As a first approach, we have decided the text interpreter output to prevail over the gesture information. In this way, when a contradictory situation occurs, the final request will be based on the text interpretation. When no contradiction exists between both sources, the gesture information is used either to confirm the text interpretation (redundant information), or to complete it (complementary information). For instance, using both voice and a gesture to stop a specific action provides redundant information through both channels. In contrast, using voice to determine an action and a gesture to indicate the location of the object that should suffer that action provides complementary information through both channels. In the second case, the knowledge base is used to check if the gesture information makes sense for a given task, discarding incoherent frame generation.

As a result, the fusion engine will send to the executive manager the potential, coherent and reliable requests that are understandable for the robot. The executive manager will then be in charge of task-planning issues considering those potential requests.

Case Study

In the context of the FourByThree project in which the work presented here is inscribed, there are several industrial scenarios that include human-robot collaboration via natural communication. For an initial validation of the semantic multimodal interpreter, we have selected a scenario that involves two such collaborative tasks that are carried out via interaction between a person and a robot. One task involves the collaborative assembly/disassembly on the same dies, handling different parts of the dies and (un)screwing bolts as required. The other task involves a collaborative deburring operation of wax patterns that requires managing different parts adequately in order to build a mould.

In the case of assembly task, the human and the robot work independently (un)screwing bolts on different parts of the die, and then they work together simultaneously (un)screwing different bolts on the same die cover. For the deburring activity, the human and the robot perform sequential tasks on the same workpiece in a sychronized manner, where the person glues and positions parts on the workbench while the robot deburrs them.

Considering these two contexts, we have identified the possible tasks the robot can fulfill and we have created a knowledge base starting from the knowledge manager ontology. We have also included in the knowledge base the elements that take part in both processes, together with the relations they have with respect to the tasks.

We have simulated the robot initialization to check for correct functionality. Currently we are carrying out a laboratory experimentation for evaluating the performance of the multimodal semantic interpreter.

Conclusions and future works

We have presented a semantic driven multimodal interpreter for human-robot collaborative interaction focused on industrial environments. The interpreter relies on text and gesture recognition for request processing, dealing with the analysis of the complementary/contradictory aspects of both input channels, taking advantage of semantic technologies for a more accurate interpretation due to the reasoning capabilities it provides.

This approach is generic and it can be applied in different industrial scenarios. However, in order to evaluate the approach, we are working on a specific scenario that includes the human-robot collaborative activities of assembling and deburring. We intend to measure the whole system accuracy as well as the benefit of a multimodal system against a mono-modal one in industrial environments. In addition, we will assess the usability and the benefits of such a system in industrial scenarios, as part of the advancement towards natural communication in human-robot collaborative work.

Acknowledgment

The FourByThree project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No. 637095

References

Bannat, A.; J. Gast, T. R.; Rösel, W.; Rigoll, G.; and Wallhof, F. 2009. A multimodal human-robot-interaction scenario: Working together with an industrial robot. 303–311.

Burger, B.; Ferrane, I.; and Lerasle, F. 2010. Towards multimodal interface for interactive robots: Challenges and robotic systems description.

Di Marco, D.; Tenorth, M.; Hussermann, K.; Zweigle, O.; and Levi, P. 2013. Roboearth action recipe execution. 117–126.

Fong, T.; Illah, R.; and Dautenhahn, K. 2003. A survey of socially interactive robots. 143–166.

Gonzalez-Agirre, A.; Laparra, E.; and Rigau, G. 2012. Multilingual central repository version 3.0: upgrading a very large lexical knowledge base.

Goodrich, M., and Schultz, A. 2007. Human-robot interaction: A survey. 203–275.

Gunhee, K.; Woojin, C.; Munsang, K.; and Chongwon, L. 2004. The autonomous tour-guide robot jinny. 3450–3455.

Kollar, T.; Tellex, S.; Roy, D.; and Roy, N. 2010. Toward understanding natural language directions. 259–266.

MacMahon, M.; Stankiewicz, B.; and Kuipers, B. 2006. Walk the talk: Connecting language, knowledge, and action in route instructions.

Padró, L., and Stanilovsky, E. 2012. Freeling 3.0: Towards wider multilinguality.

R. Stiefelhagen, C. Fugen, P. G. H. H. K. N., and Waibel, A. 2004. Natural human-robot interaction using speech, head pose and gestures. 2422 – 2427 vol.3.

Rossi, S.; Leone, E.; Fiore, M.; Finzi, A.; and Cutugno, F. 2013. An extensible architecture for robust miultmodal human-robot communication. 2208–2213.

Tenorth, M.; Kunze, L.; Jain, D.; and Beetz, M. 2010. Knowrob-map - knowledge-linked semantic object maps.

Thrun, S.; Bennewitz, M.; Burgard, W.; Cremers., A.; Dellaert, F.; Fox, D.; Hähnel, D.; Lakemeyer, G.; Rosenberg, C.; Roy, N.; Schulte, J.; Schulz, D.; and Steiner, W. 1999. Experiences with two deployed interactive tour-guide robots.

Wang, T., and Chen, Q. 2011. Object semantic map representation for indoor mobile robots. 309–313.

Winograd, T. 1971. Procedures as a representation for data in a computer program for understanding natural language.

Demonstration of Complex Task Execution using Basic Functionalities: Experiences with the Mobile Assistance Robot, “ANNIE”

Christoph Walter, Erik Schulenburg, José Saenz, Felix Penzlin, Norbert Elkmann

Dept. of Robotic Systems

Fraunhofer Institute for Factory Operation and Automation (IFF)

Magdeburg, Germany

<firstname>.<lastname>@iff.fraunhofer.de

Abstract

Mobile assistance robots, also known as mobile manipulators, are slowly making their way out of research laboratories and into real world applications in industry. In this paper we will describe the mobile assistance robot, “ANNIE”, which was developed by the Fraunhofer IFF Business Unit Robotic Systems and features a holonomic motion platform, a KUKA lightweight arm LBR 4+ outfitted with a universal gripper, and a wide array of integrated sensors including tactile sensors and a light-field camera. After a brief overview of the robot hardware, we will describe three exemplary tasks carried out by ANNIE, namely tightening screws in a mechanical assembly, acting as a third-hand to assist a human operator lift and position a long, unwieldy part, and autonomously carrying out intra-production logistic tasks (pick and place). The skills-based programming system and the background services necessary for integration will also be described in this paper.

Introduction

In this paper we will describe in detail three use-case scenarios carried out by the mobile assistance robot, “ANNIE”, which was developed by the Fraunhofer IFF Business Unit Robotic Systems. This paper will begin by explaining the motivation for using mobile assistance robots in production today and briefly reviewing the state of the art for mobile assistance robots for industrial applications. Following a description of the ANNIE hardware, three use-cases featuring ANNIE will be presented, namely tightening screws in a mechanical assembly, acting as a third-hand to assist a human operator lift and position long, unwieldy parts, and autonomously carrying out intra-production logistic tasks (pick and place). We will finish by describing the skills-based programming system and the background services necessary for integration. Finally we will provide a brief outlook on future activities for achieving an increased level of capability with ANNIE.

Figure 1: Mobile Platform “ANNIE” - Overview of major hardware components of the mobile assistance robot

Motivation for mobile assistance robots

Mobile assistance robots, also known as mobile manipulators, are currently the focus of a large amount of research and development efforts, both in industry and academia. In contrast to traditional industrial robots, which are bolted to the ground and are programmed to repeat one task a large number of times, mobile robots offer an almost unlimited workspace and are designed to be flexible enough to allow for efficient use even with changing tasks and lower amounts of repetition of single tasks. Recent advances in navigation, safety, and machine vision have all reached a level of maturity which allow the use of mobile platforms in conjunction with a robotic manipulator.

From an industrial point of view, mobile assistance robots, which work in close contact with human workers and are collaborative by nature, present a natural solution to a wide variety of challenges in production today. On the one side, demographic change is increasing the average age of the workforce, and robots [1] which can support heavy loads or carry out repetitive tasks offer a chance for workers to focus on cognitive, value-added tasks and remain active and healthy on the shop floor for as long as possible. On the other side, the trend towards increasing levels of customization lead to a demand for more flexible production systems. Seen within the context of other industrial ICT trends (increasing levels of digitalization, Internet of Things, Industry 4.0, etc.) there are a number of technologies currently reaching a level of maturity which will soon allow for the effective and economic use of mobile manipulators in production. Nevertheless, in order to reach their full potential in terms of economic effectiveness, flexibility, and ultimately industrial relevance, mobile assistance robots need to be universal in nature and provide strong tools for system integration, programming of complex tasks, and autonomy.

State of the Art

While there are a wide range of mobile robots [2] ranging from autonomous guided vehicles (AGVs), tele-operated systems to autonomous cars, for the purposes of this paper we will focus on mobile robots featuring an articulated robotic arm and which are intended for use in industrial environments. Such robots are collaborative in nature, as they are intended to work side-by-side with humans without separating fences. Commercially-available mobile robots include KUKA KMR iiwa [3], Neobotix family of mobile manipulators [4] and PAL Robotics TIAGO [5]. Both commercial systems face limitations either due to their price or low payload capability. A further mobile manipulator of industrial relevance, which has until now only been described as a proof of concept and not as a productive system, is the FAUB [6] system developed by KUKA Systems for Boeing. This system features a high payload, traditional industrial robot mounted on an omniMove platform for carrying out assembly tasks in aerospace production. While the use of high payload robots is interesting, this system has not been explicitly been described as collaborative in nature. In conclusion, while many specific technologies necessary for mobile assistance robots have been the focus of research for decades, there are relatively few commercially-available products.

We believe that this relatively low number of commercially-available products is a reflection of the challenges currently faced in developing a tightly integrated system featuring a large number of heterogeneous technologies which satisfies industrial requirements of performance, efficiency, and ease-of-use. Figure 2 demonstrates the design space [7] available to robotics developers, whereby movement in a positive direction along each individual axis is a proxy for increased complexity and cost. This diagram thus helps to underscore the challenges faced by developers of mobile, assistance systems for use in complex applications. Indeed, recent, high-profile publicly-funded research projects featuring mobile assistance robots including TAPAS [8], VALERI [9], CARLoS [10] and ISABEL [11] have focused not only on individual technologies (e.g. safety, human-robot collaboration, etc.), but also on methods for increased performance and easier programming.

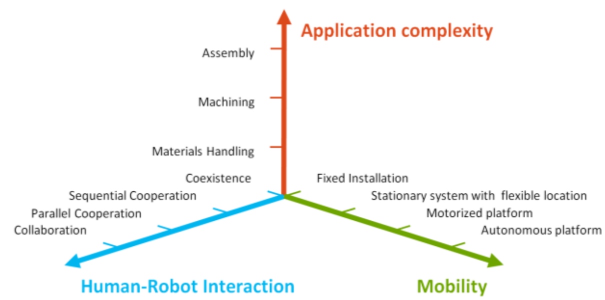


Figure 2: Design space for robotics developers in terms of mobility, application complexity and human-robot collaborative ability.

Description of "ANNIE"

The mobile assistance robot, ANNIE was developed by the Fraunhofer IFF in Magdeburg, Germany as a universal system able to be used for a large number of industrially relevant applications. The robot was intended as a development platform, both for showcasing individual technologies and for test and development of the necessary software and systems integrations tools necessary for supporting such a universal system. The mobile assistance robot consists of an omni-directional platform equipped with a lithium-ion battery capable of powering up to 12 hours of continuous robot activity. The actuated torso was designed to allow for a low center-of-gravity during high speed navigation and also to allow for a large end-effector workspace. The dimensions are such that the gripper can reach objects laying on the ground and up to 2.10 m in height. A LBR 4+ articulated robot arm was chosen for its sensitivity and force-feedback capabilities. The manipulator is outfitted with a manual tool changer and currently features a Robotiq 3-finger hand outfitted with tactile sensors custom built by the Fraunhofer IFF [13]. A foveated, hybrid light-field camera system based on the one described in [11] is mounted above the torso. Its field of view is enhanced by a pan-tilt unit. Additionally, this sensor head contains a long tele zoom camera in addition to a Microsoft Kinect. ANNIE features a number of different on-board sensors for navigation and localization. In addition to two LIDAR sensors for localization and mapping, a fixed camera situated at the rear end of the mobile platform can be used for visual localization [12].

The overall dimensions of the platform were designed not only to allow for the robot to carry out the largest possible number of industrial applications, but also to allow for access to workspaces originally designed for humans. Thus the height and width of the platform were designed to allow passage through standard door frames.

The overall design of the platform makes it suitable for a wide range of research questions.

Use-Case Scenarios with ANNIE

In the following we will describe three use-case scenarios carried out with ANNIE, whereby basic functionalities were used for fast and effective programming. These use-cases were chosen due to their industrial relevance and to showcase the wide range of capability of mobile assistance systems. The implementation of these use-cases requires many individual skills. All skills are implemented as software modules in a dedicated software framework. They offer reusable functional blocks which are parameterized for the concrete scenarios. The software framework in turn manages communication between software modules and offers basic functionalities for common services.

Use-Case 1 - Assistance

The first use-case scenario is a collaborative assembly task. A worker has to attach a long structural element to a rack. The robot assists with carrying the part and holds one end in place while the human screws the other end to the frame as shown in Figure 3.

In the beginning of the procedure several long parts are located in a fixture on a table. The human worker starts the process via a voice command to get the attention of the robot. This is achieved by speech recognition software and predefined phrases. As a result the robot turns its head in the direction of the speaker and verifies his identity by face recognition. Alternatively the worker can carry a tablet computer and use touchscreen input on a dedicated web interface to communicate with the robot. He can also use a headset for speech input if the noise level of the environment demands it.

Figure 3: Demonstration of Assistance Use-Case

After the process has been initiated, the robot starts tracking the posture of the person using standard skeletal tracking software of the Microsoft Kinect. The human being can then choose the actual part to pick-up by gesture control. ANNIE lifts one end of the part and moves to the assembly position. The worker guides the other end and can control the height and adjustment of the part within a certain range. For this purpose, the robot measures force and torque while holding the element with the internal sensors of the KUKA lightweight arm. When the worker has mounted his end of element to the frame he grabs the other end that is still held by the robot. Triggered by a voice command the robot performs some checks if it is actually safe to release the part and hand it over to the human. If these checks are successful the gripper releases the part, the robot arm is retreated and the mobile platform moves away to make room for the worker to finish the assembly.
This description reveals many distinguishable skills demanded from ANNIE. These skills include:

- speech recognition and sound location,
- gesture recognition (skeletal tracking),
- face detection and recognition,
- gripping of pre-defined parts,
- self-localization, path planning and path following,
- force/torque guided movement,
- handing-over of parts to worker.

Each skill is represented by a functional block within the aforementioned software framework. Parameterization and composition of those functional blocks to implement an entire scenario will be described in the section on our skill-based programming system.

According to the categories established in Figure 2, this use-case can be characterized as featuring collaboration-type human-robot interaction for an assembly operation featuring an autonomous mobile robot with a restricted amount of mobility needed to complete the task.

Use-Case 2 – Assembly

The second scenario involves ANNIE tightening pre-assembled screws in an assembly. The idea is to relieve the worker from this tedious and time-consuming process. The human only has to insert the screws into the holes with one or two turns and command the robot to tighten them.

To allow for a high degree of flexibility, ANNIE is equipped with a flexible 3-finger robotic hand and can more or less make use of conventional tools. In this case we modified a standard cordless electric screwdriver with an Intel Edison module to allow control of the screwdriver via Wi-Fi commands. To fulfill the described task, ANNIE picks up this screwdriver from a tool holder and orients its head towards the construction part (Figure 4). It should be noted that the tool holder is not a typical fixture for industrial robots, which normally sub-millimeter positioning accuracies, but rather in a holder more suitable for humans, where the position of the screwdriver can vary in the range of a several millimeters. Although the positions of the screws are known a-priori from a CAD model, there are nevertheless several uncertainties. On one hand, the position of the mobile platform relative to the part might be imprecise. On the other, the grasp of the tool is different each time, as the tool is not in a fixed position and also to the previously mentioned uncertainty. To compensate for these uncertainties, we employ the light-field camera system to estimate both the part and the hand and based on that locate the tool tip relative to the screw.

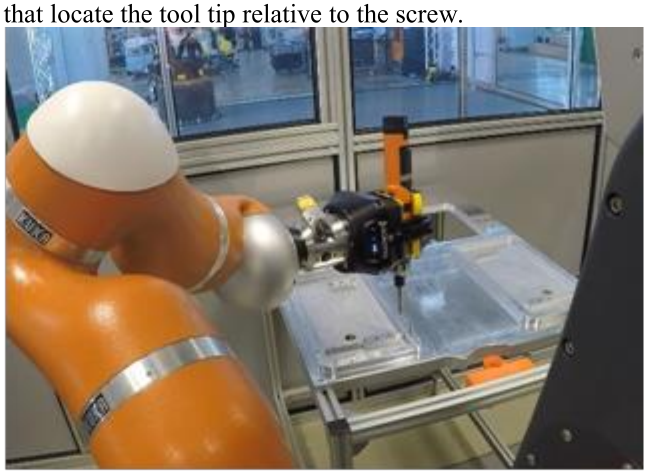


Figure 4: Demonstration of Assembly Use-Case

The light-field camera on ANNIE'S head is used to determine the position of the M5 screw heads utilized in our test case and the tip of the screwdriver. While tracking both positions, the tool is inserted into the screw head. By observing the force feedback with the internal sensors of the KUKA lightweight robot arm, we can verify the success of this procedure. The screwdriver is then turned on until a certain torque is reached. The robot then repeats this process with the following screws.

This scenario requires different robot skills than assistance scenario:

- grasping of tools (including verification of grip),
- identification of small parts,
- visual localization of parts and robot,
- controlling external tools.

According to the categories from Figure 2, this use-case can be characterized as featuring sequential cooperation for an assembly operation featuring an autonomous mobile robot with a low amount of mobility needed to complete the task. This assembly task is particularly complex due to challenging requirements regarding object localization and error detection.

Use-Case 3 – Logistics

In production-based materials transfer we see a broad range of applications. Here, parts not only need to be transported from one location to the other, there may also be some handling skills required at the different locations. Applications range from specialized transport tasks in highly structured production environments to co-worker scenarios which require more diverse and flexible skills to cope with challenges presented by an environment primarily designed for human personnel. One such scenario is the autonomous transport of items in a life-science lab.

This third scenario represents the most challenging in terms of the variety of required skills. In the life-science setting we usually find standardized objects like multiwell or microwell plates that contain the samples. Depending on the process, these objects go through different machines like analyzers or incubators. Machines may be combined into islands of automation with integrated handling solutions and dedicated load-ports. Furthermore, stations may be situated in separate rooms.

For the life-science scenario we assume that the robot must be able to autonomously navigate to different rooms, including being able to open a standard door. It needs to pick up microwell plates from a table that were prepared by a human lab assistant. We want to collect up multiple plates and put them into a larger carrier for efficiency. Next, the plates need to be taken to a drop-off point representing the load-port of some automated handling machine. These requirements would usually call for different grippers specific to the objects that require handling (plates, carrier, door handle). However, we were able to successfully use the 3-finger robotic hand for all handling and manipulation tasks in this scenario.

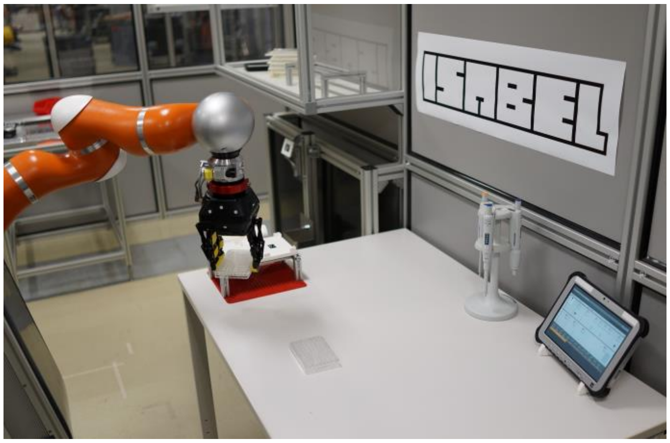


Figure 5: Demonstration of Logistics Use-Case

In more detail, the process involves the following steps: Entering the room involves opening the door by its handle as well as navigating through the narrow door opening. Path planning within the room is used to navigate to the table where the manually prepared samples need to be collected. An empty carrier is then taken from a storage shelf next to the table and placed into a rig on the table. The robot then searches for microwell plates on the table. These plates are made of glossy transparent plastic and present a challenge for commodity 2½D sensors like the Kinect. In [11] we presented an approach to use a light field camera array in such situations.

The robot uses motion planning with object specific, pre-defined grip poses to pick up the plates. When inserting the plates into the carrier the process required a specific sub-routine which implements a particular strategy for putting both parts together using force guidance.

The final steps of the process involve picking up the carrier by its handle and navigating to the drop-off location. After putting the carrier down at that location the process is finished and the robot is available for further tasks.

Use-case 3 can also be categorized according the Figure 2. The scenario requires a high degree of autonomy on the mobility axis. Regarding human-robot collaboration the scenario can be seen as parallel cooperation, since the operator and the robot share the same workspace and parts. We consider the complexity of materials handling tasks to be at a medium level.

Skill-Based Programming System

From an end user perspective, programming the system means defining a process as a sequence of skills. In our case we chose commercially available speech and gesture interaction techniques as a capable method for parametrization of actions during runtime. Current methods for the description of a complex process on an abstract level are considered very time-consuming and do not offer the possibility for the worker to get a quick overview of a saved process. A widely accepted and intuitive input method that fulfills industrial requirements is the use of touch interfaces. The target group of workers is generally considered to be familiar with the concept of touch interfaces for human-computer interaction.

While the core concept of programming the system from a user perspective means defining a sequence of skills, from a system perspective, each skill consists of a complex program that can be parametrized in various ways, if need be, during runtime. In imperative text-oriented programming languages it would be expressed as a sequence of commands, each command in a new line. That means a sequence is read top to bottom. For non-programmers it is more intuitive to read from left to right, according to roman script. In consideration of this habit, the actions are arranged from left to right.

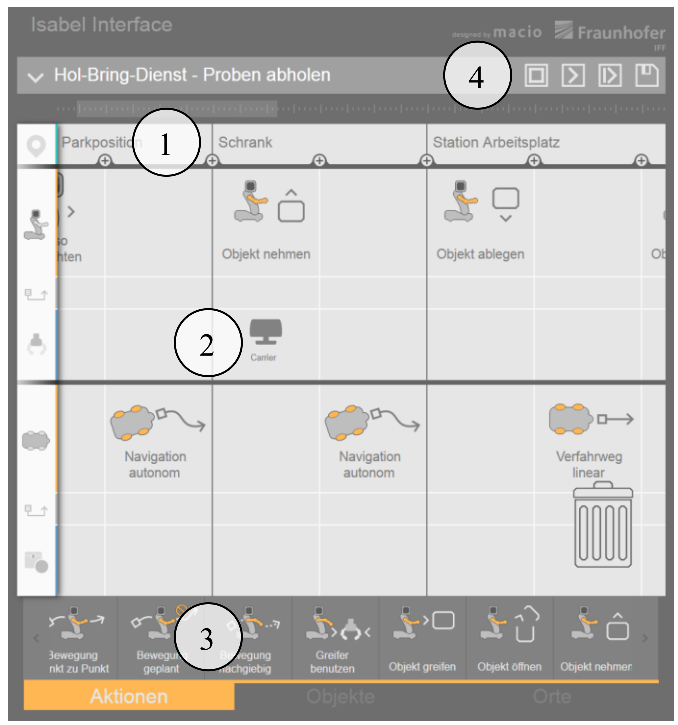


Figure 6: Touch user interface

The metaphor of the programming environment is inspired by video sequence editors. As shown in Figure 6, the interface offers multiple tracks to the user. The upper track (1) defines the location where the actions take place. Beneath this, the actions themselves are defined. There are two tracks concerned with actions, namely actions that use manipulation skills and actions that involve navigation of the mobile platform. However, this was done only to give the program an easier to read structure. It is not possible to have two skills in the same timeslot. Underneath each action, the corresponding track objects are defined (2). Skills are represented by icons (3). These icons can be dragged on the corresponding track by a touch gesture. From a system developer perspective, each action corresponds to a small script. These scripts are programmed in an extended version of LUA on a high abstraction level.

Acceptance of the system highly depends on transparent decisions. If the worker is not able to recognize the current action and anticipate the next steps, a collaboration is complicated. The current state of the process is shown by the same visual representation (see Figure 7). The executing action is slightly lifted and rotated (1). We do not want to demand that the end-user to consider error handling in detail, so each execution block deals with error handling internally. In exceptional cases, if an error cannot be solved, fault recovery requires an adaption of the process. During normal operation, actions are executed from left to right, but if the result of an action is somehow not correct, the user can define to jump to a specific action. Jump options are visualized by arrows (2).

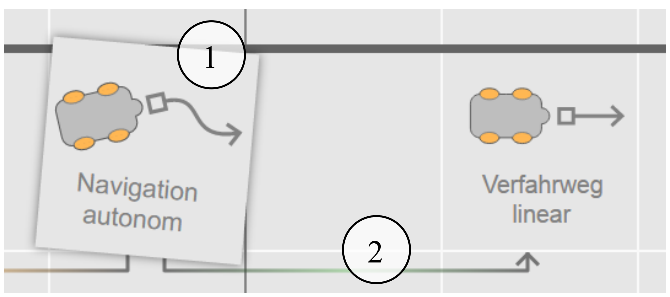


Figure 7: Execution status

Each action can therefore define multiple final states that are returned by the LUA script. In addition to these final states, each action defines a list of optional parameters, which can be viewed and changed by the user on the interface as instances of skills. Objects and locations are considered as implicit parameters for the current skill. The actual execution of the skill depends on the parameters. Strategies specific to the handled object are chosen. For example there are different strategies to pick up different types of objects. The user doesn't have to consider optimal gripping positions or trajectories. He defines the action and the object. The choice of the optimal subskill is done by the system without any effort of the user. Only a correct combination will lead to flawless process execution. The interface only allows the insertion of useful combinations and prompts the user to add mandatory elements.

Background Services

While skills or high-level function blocks are implemented using an extended version of LUA, lower-level functions such as various planning, execution, or sensing capabilities are implemented as loosely coupled distributed software modules in a proprietary publish/subscribe system with service orientation, called the RS-Framework. Typical classes of background services with particular relevance for mobile assistance robots are: (smart) motor services to control actuators; sensor services to acquire data; perception services to extract information from sensory input; planning services for navigation and motion planning. While most services are generic, some are hardware dependent and some are dedicated to support only individual skills.

This architecture on one hand favors separation of concerns which is a highly desirable property in complex systems. However, from a control system perspective, our implementation yields a mixed-real time system. From a motion planning perspective, we not only have to deal with partial or unreliable information but also with planners that are intentionally limited in scope. An example is the planning of arm movement separate from head movement. In this particular case, a self-collision may occur during motion execution. While this is unlikely, it needs to be addressed. This is done at the execution level. We developed a scheme for augmenting functions provided by smart motor services by providing program fragments as parameters.

These programs are then executed in the control loop of the motor service and can access additional data that can be streamed from other services (see Figure 8). This enables the discovery and handling of unpredicted situations that are not only the result of imprecise planning but also of uncertainty in sensory perception as well as system timing.

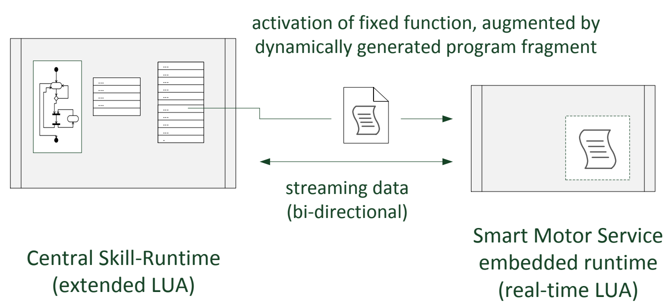


Figure 8: "Smart Motor" entity augmented by program fragments as service parameters

Overall, our architecture for background services addresses both, technical challenges in designing complex and reliable real-time systems, and the general lack of omniscience that impedes any robot's actions in loosely structured environments.

Conclusion

In this paper we introduce the mobile assistance robot, "ANNIE", which was developed as a universal system that can be used for a large number of industrially relevant applications. This flexibility is reflected though the three use-case applications, which have also been described in detail. In particular, the focus on programming the applications through skills-based techniques, including the use of a novel graphical user interface have allowed for more effective programming. Currently the individual skills are manually inserted along the timeline by an operator. In addition to the further development of individual skills to enable a larger variety of applications, collaboration with planning experts would be useful to make improved use of available semantic information to increase the autonomy in the planning and scheduling of basic skills for carrying out complex tasks.

Acknowledgement

Parts of this project (namely use-case 3 and the design of the touch interface) have received funding from the German Federal Ministry of Education and Research (BMBF) under grant agreement no. 01IM12006C (ISABEL).

References

[1] Schenk, M.; Elkmann, N. „Sichere Mensch-Roboter-Interaktion: Anforderungen, Voraussetzungen, Szenarien und Lösungsansätze.“ Demografischer Wandel-Herausforderungen für Arbeits-und Betriebsorganisation der Zukunft. GITO, Berlin, pages 109-120, 2012.

[2] Shneier, M.; Bostelman, R. "NISTIR 8022 - Literature Review of Mobile Robots for Manufacturing". National Institute of Standards and Technology. May 2015.

[3] http://www.kuka-robotics.com/en/products/mobility/KMR_iiwa/start.htm. March 16, 2016.

[4] http://www.neobotix-roboter.de/mobile-roboter-uebersicht.html. March 16, 2016.

[5] http://tiago.pal-robotics.com/. March 16, 2016.

[6] http://www.kukaaero.com/capabilities/aerospace-automation. March 16, 2016.

[7] Behrens, R.; Saenz, J.; Vogel, C.; Elkmann, N.: "Upcoming Technologies and Fundamentals for Safeguarding All Forms of Human-Robot Collaboration", 8th International Conference Safety of Industrial Automated Systems (SIAS 2015), Königswinter, Germany 18-20 November, 2015. ISBN 987-3-86423-163-6

[8] Andersen, R.S.; Nalpantidis, L.; Krüger, V.; Madsen, O.; Moeslund, T.B.. „Using robot skills for flexible reprogramming of pick operations in industrial scenarios“. In Proceedings of the 9th International Conference on Computer Vision Theory and Applications (VISAPP), volume 3, pages 678–685. SciTePress 2014.

[9] Fritzsche, M.; Saenz, J.; Penzlin, F.: "A Large Scale Tactile Sensor for Safe Mobile Robot Manipulation", Human-Robot Interaction (HRI), 11th International Conference on, New Zealand, 07.-10. March, 2016

[10] www.carlosproject.eu. March 16, 2016.

[11] Walter, C.; Penzlin, F.; Schulenburg, E.; Elkmann, N. "Enabling multi-purpose mobile manipulators: Localization of glossy objects using a light-field camera". In: IEEE 20th Conference on Emerging Technologies and Factory Automation (ETFA) 2015: 1-8

[12] C. Walter, E. Schulenburg, M. Poggendorf and N. Elkmann, "A QoS enabled visual sensor-system applied to the problem of localizing mobile platforms in indoor environments," Sensors, 2011 IEEE, Limerick, 2011, pp. 1804-1807

[13] Fritzsche, M., Elkmann, N., "Tactile sensor with indepenent sensor cells," Patent DE 10 2007 022 871 A1, filed May 14, 2007.

[14] Hanses, M.; Walter, C.; Lüder, A..: "Operating articulated objects with force sensitive mobile manipulators." In: IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA), 2015. S. 1-4.

Real-Time Obstacle Avoidance for Continuum Manipulator: Towards Safer Application in Human Environments

Ahmad Ataka, Ali Shafti, Ali Shiva, Helge Wurdemann, and **Kaspar Althoefer**
Centre for Robotics Research, King's College London
Strand, London, WC2R 2LS
England, United Kingdom

Abstract

The rigid-link manipulators have been extensively employed in industrial setting nowadays. However, problem arises when such manipulators are assigned to work alongside human or employed in cluttered, complex, and tight environment. Hence, continuum manipulator, with its inherent compliant and flexibility, is preferable in this aspect. In this paper, we present our recent work presenting a real-time pose estimation and obstacle avoidance for continuum manipulator in dynamic environments. The approach is shown to work in a model of both single-segment and multi-segment continuum manipulator and also in a real tendon-driven single-segment continuum manipulator in dynamic environment, and thus, suitable to be used in human environments.

Introduction

Nowadays, the field of rigid-link manipulators is a well-established discipline. Its ability to have precise position control and trajectory generation for doing various manipulations and tasks has made it popular in industrial setting. The rigid-link manipulators are now used daily in various industrial environment and manufacturing plants worldwide. However, the current generation of rigid-link manipulators are mainly employed automatically in task with limited human intervention due to the safety reason. This constraints the applicability of such manipulators to tasks which require human-robot collaboration.

Several researchers proposed solutions for safer human-robot interaction, such as the utilization of flexible joint (Ozgoli and Taghirad 2006) and variable stiffness actuator (Kim and Song 2010), an elastic strip approach which exploits redundancy of the manipulators (Khatib et al. 1999), (Brock and Khatib 2002), and others which include safety criteria to robots motion planning and obstacle avoidance (Haddadin et al. 2013). These, however, do not eliminate the fact that most industrial manipulators still need an open space and well-defined environment in order to execute the task.

One alternative solution is by employing a continuum-style manipulators, mainly used in medical applications (Burgner-Kahrs, Rucker, and Choset 2015). Mostly inspired by nature, such as octopus arm (McMahan et al. 2006) or snake (Hirose 1993), continuum manipulator has an ability to bend at any point along its body. Thus, it is very suitable to be used in complex and tight environments where the rigid-link counterparts would be unable to manoeuvre. Its inherent compliance will also make it safer and more adaptable when interact with sensitive environment, including human. Continuum manipulators, for example mounted on mobile platforms, will have more flexibility than their rigid-link counterparts when employed in industrial setting with human presence.

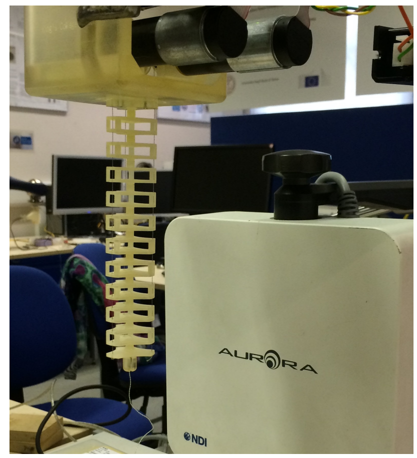


Figure 1: A tendon-driven single-segment arms used as a model in this paper. To measure the tip's pose and the obstacle's pose, the NDI Aurora tracker can be adopted.

However, the advantage of backbone flexibility possessed by continuum manipulators comes with the consequence of difficulty in modelling their structure. This will in turn complicates the motion planning and trajectory generation. The model-based pose estimator is needed to correctly estimate the pose of manipulators body such that the real-time obstacle avoidance can be employed to safely avoid collision with human or dynamic environments.

In this paper, we present a real-time obstacle avoidance for continuum manipulators in dynamic environments. The obstacle avoidance is equipped with non-linear observer to estimate the pose of any point along the body of manipu-

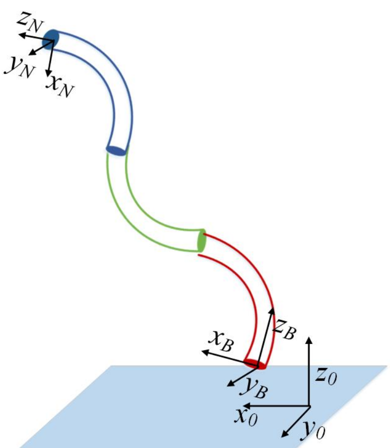


Figure 2: An illustration of three segments continuum manipulator with mobile base.

lator to make sure that the whole body of the manipulator can avoid collision. The algorithm is tested for a model of tendon-driven single-segment continuum manipulator as shown in Figure 1 and has been verified experimentally as presented in (Ataka et al. 2016). The obstacle avoidance is also implemented to a model of multi-segment continuum manipulators with mobile platform (Ataka et al. 2016). The overall algorithm is shown to work well in avoiding moving obstacle, and thus, makes it suitable to be used in human environments.

Continuum Manipulator

Kinematic Model

Here, we present a model of a tendon-driven continuum manipulator. The manipulator is moved by modifying the length of three uniformly distributed tendons along the surface in each segment The model is based on constant-curvature assumption, i.e. each segment is assumed to have a constant radius of curvature at a given time. Each segment can then be parameterized by configuration space variables $\mathbf{k}_i$ consisting of curvature κ_i, deflection angle ϕ_i, and segment length s_i. The forward kinematics relation mapping these variables to task space position of the segments tip is presented in (Webster and Jones 2010). For a continuum manipulator with mobile platform, as shown in Figure 2, the homogeneous transformation of the end-effector with respect to the world frame is expressed as

$$ {}_0^N\mathbf{T}(\mathbf{k}) = \mathbf{T}_B \prod_{i=1}^{N} {}_{i-1}^{i}\mathbf{T}(\mathbf{k}_i) \tag{1} $$

Where N specifies the segment number, $\mathbf{k} = [\mathbf{k}_1 \quad \mathbf{k}_2 \quad \dots \quad \mathbf{k}_N]^T$ denotes the vector consisting of configuration space variables of all segments, and $\mathbf{T}_B \in SE(3)$ denotes the homogeneous transformation matrix of the frame attached to the base.

Moreover, the mapping from the configuration space variables $\mathbf{k}$ to the actuator space $\mathbf{q}$, in this case specifies tendons length, is also well defined in (Webster and Jones 2010). Adding the mobile platform pose $\mathbf{q}_0$ to the actuator space $\mathbf{q}$, we have $\mathbf{q} = [\mathbf{q}_0 \quad \mathbf{q}_1 \quad \dots \quad \mathbf{q}_N]^T$ where each component $\mathbf{q}_i$ consists of the tendon length of each segment, written as $\mathbf{q}_i = [l_{i1} \quad l_{i2} \quad l_{i3}]^T$.

In order to specify any point along the body of the manipulator, we use a scalar $\xi_i \in [0,1]$ for each segment, ranging from the base ($\xi_i = 0$) to the tip ($\xi_i = 1$). The list of scalars ξ_i for all segments are then combined to be a vector ξ whose value is govern by $\xi = \{\xi_r = 1 : \forall r < i, \xi_i, \xi_r = 0 : \forall r > i\}$.

In short, we can write the forward kinematics of the continuum manipulators as

$$ {}_0^N\mathbf{T}(\mathbf{q},\xi) = \begin{bmatrix} \mathbf{R}(\mathbf{q},\xi) & \mathbf{p}(\mathbf{q},\xi) \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{2} $$

where $\mathbf{R}(\mathbf{q},\xi) \in SO(3)$ stands for the rotation matrix and $\mathbf{p}(\mathbf{q},\xi) \in \mathbb{R}^3$ stands for the position vector of the point along the body of manipulator. The Jacobian for our kinematic model, defined as $\mathbf{J}(\mathbf{q},\xi) = \frac{\partial \mathbf{p}(\mathbf{q},\xi)}{\partial \mathbf{q}} \in \mathbb{R}^{3\times(3N+6)}$, is expressed as follows

$$ \dot{\mathbf{p}}(\mathbf{q},\xi) = \mathbf{J}(\mathbf{q},\xi)\dot{\mathbf{q}} \Leftrightarrow \dot{\mathbf{q}} = \mathbf{J}(\mathbf{q},\xi)^{-1}\dot{\mathbf{p}}(\mathbf{q},\xi). \tag{3} $$

State-Space Representation

The kinematic model can also be expressed in terms of state space representation, i.e. the state equation and output equation. Here, we present the state space analysis for a static single segment only. We use the tendon length $\mathbf{q} \in \mathbb{R}^3$ as our state, $\mathbf{x} \in X$. Hence, the input to our system, $\mathbf{u} \in U$, is given by the actuator lengths rate of change, $\dot{\mathbf{q}} \in \mathbb{R}^3$, which is governed by DC motors connected to the tendon. The measurement value $\mathbf{y}_k \in Y$ comes from a position sensor embedded in the tip of manipulator. The NDI Aurora tracker, like the one shown in Figure 1, can be used for this purpose. Hence, the output equation matches the component of the matrix given by the forward kinematics relation in 2, $\mathbf{p}(\mathbf{q},\xi)$ for $\xi = 1$ (tip). Here, X, U, and Y denote the state space, input space, and output space respectively.

The state space and output equation in discrete form, for a sampling time of Δt, can then be expressed as

$$ \mathbf{x}_{k+1} = f(\mathbf{x}_k,\mathbf{u}_k) = \mathbf{x}_k + \Delta t \mathbf{u}_k. \tag{4} $$

$$ \mathbf{y} = g(\mathbf{x}_k) = \mathbf{p}(\mathbf{q},\xi = 1), \tag{5} $$

where the function $f : X \times U \to X$ is used to map the current state and input to the next state while $g : X \to Y$ is used to map the current state to the output.

However, using only this information is not enough to estimate the pose of the whole body of the manipulator, due to the unknown initial state value. This is where the non-linear observer is needed to estimate the state and, in turn, use the estimation state to estimate the pose of any point along the body of manipulators.

Pose Estimation and Obstacle Avoidance

Pose Estimation

Based on the presented state space model, we employ a well-known Extended Kalman Filter (EKF) approach to our

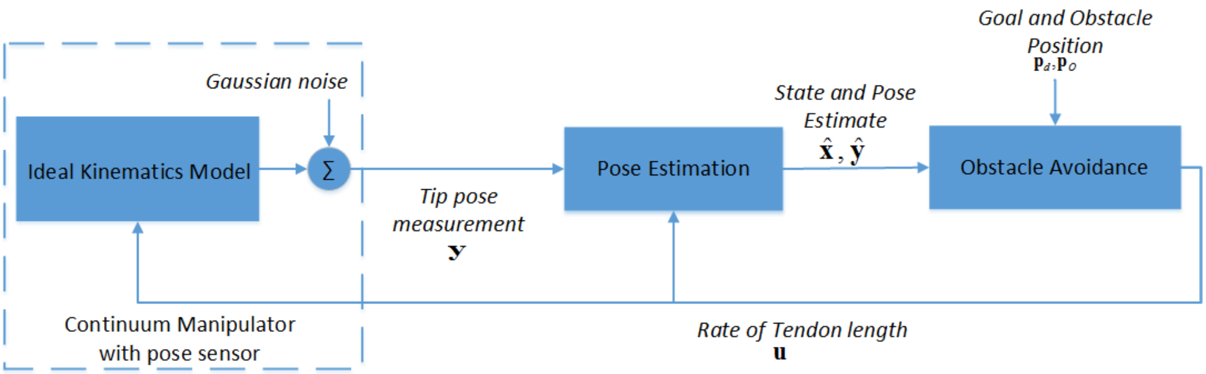


Figure 3: The proposed pose estimator and obstacle avoidance algorithm. An ideal kinematics model, added with Gaussian noise, is used to replace the continuum manipulator and the pose sensor during the simulation.

model. The EKF can be formulated as

$$\begin{aligned} \hat{\mathbf{x}}_{k+1|k} &= f(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_k), \\ \mathbf{P}_{k+1|k} &= \mathbf{A}_k \mathbf{P}_{k|k} \mathbf{A}_k^T + \mathbf{Q}_k, \\ \mathbf{K}_k &= \mathbf{P}_{k+1|k} \mathbf{C}_k^T (\mathbf{C}_k \mathbf{P}_{k+1|k} \mathbf{C}_k^T + \mathbf{R}_k)^{-1}, \\ \hat{\mathbf{x}}_{k+1|k+1} &= \hat{\mathbf{x}}_{k+1|k} + \mathbf{K}_k (\mathbf{y}_k - g(\hat{\mathbf{x}}_{k+1|k})), \\ \mathbf{P}_{k+1|k+1} &= (\mathbf{I} - \mathbf{K}_k \mathbf{C}_k) \mathbf{P}_{k+1|k}. \end{aligned} \tag{6}$$

$\hat{\mathbf{x}}_{k+1|k+1}$, $\hat{\mathbf{x}}_{k|k}$, $\mathbf{u}_k$, and $\mathbf{y}_k$ represent the next state estimation, the current state estimation, the input signal, and the measurement data respectively. The matrix $\mathbf{Q}_k \in \mathbb{R}^{3\times3}$ and $\mathbf{R}_k \in \mathbb{R}^{3\times3}$ are the process noise variance and measurement noise variance respectively.

The matrix $\mathbf{A}_k$ and $\mathbf{C}_k$ are defined as $\mathbf{A} = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k)}{\partial \mathbf{x}_k}$, $\mathbf{B} = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k}$ and $\mathbf{C} = \frac{\partial g(\mathbf{x}_k)}{\partial \mathbf{x}_k}$ respectively and can be written as

$$\mathbf{A}_k = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k)}{\partial \mathbf{x}_k} = \mathbf{I} \in \mathbb{R}^{3\times3}, \tag{7}$$

$$\mathbf{C}_k = \frac{\partial g(\mathbf{x}_k)}{\partial \mathbf{x}_k} = \frac{\partial \mathbf{p}(\mathbf{q}_k, \xi = 1)}{\partial \mathbf{q}_k} = \mathbf{J}(\mathbf{q}_k, \xi = 1). \tag{8}$$

The estimation state can then be used to estimate the pose of any point along the body of manipulator using a forward kinematics relation in (2) by modifying the scalar ξ from 0 (base) to 1 (tip). This information is used in the obstacle avoidance stage.

Modified Potential Field

The reactive potential field presented in (Khatib 1985) is modified here. The idea is to use the potential function U to attract the tip of manipulator to a desired target position and repel its body from colliding with environment. A standard potential field, which usually produces a task space force $\mathbf{F} = -\nabla U$, is modified such that it is suitable to be used in continuum manipulators kinematic model. Rather than force, the task space velocity $\dot{\mathbf{p}}$ is produced.

The attractive potential field is given by

$$\dot{\mathbf{p}}_{\mathbf{p}_d} = -c(\mathbf{p} - \mathbf{p}_d). \tag{9}$$

where $\mathbf{p}_d$ and c represent desired position and a positive constant gain respectively. The repulsive field is expressed as

$$\dot{\mathbf{p}}_{\mathcal{O}} = \begin{cases} \eta\left(\frac{1}{\rho} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2}\frac{\partial \rho}{\partial \mathbf{p}} & \text{if } \rho < \rho_0 \\ 0 & \text{if } \rho \geq \rho_0 \end{cases}. \tag{10}$$

where $\rho = \sqrt{(\mathbf{p} - \mathbf{p}_{\mathcal{O}})^T(\mathbf{p} - \mathbf{p}_{\mathcal{O}})}$ denote the closest distance from an obstacle to the manipulator's body, η is positive constant, and ρ_0 indicates the limit distance of the potential influence.

In order to achieve safer obstacle avoidance, the repulsive potential needs to be applied not only at the tip but also at points along the body of manipulator. Therefore, we define *point subjected to potential* (PSP) as a point in the backbone of manipulator in which the repulsive potential is possible to be applied. The PSP to be chosen is the closest one to the obstacles or environments. The position of this PSP as well as the tip are all estimated by the pose estimation stage at every iteration as follows

$$\hat{\mathbf{p}}_k(\xi) = \mathbf{p}(\hat{\mathbf{x}}_{k|k}, \xi). \tag{11}$$

Finally, each attractive and repulsive velocity in task space, using their corresponding working points, are mapped to the actuator space via inverse Jacobian relation. The combined velocity in actuator space is then fed as an input to our manipulator, $\mathbf{u}_k$, as follows

$$\mathbf{u}_k = \dot{\mathbf{q}} = \mathbf{J}_e^{-1}\dot{\mathbf{p}}_{\mathbf{p}_d} + \mathbf{J}_a^{-1}\dot{\mathbf{p}}_{\mathcal{O}}, \tag{12}$$

where $\mathbf{J}_e$ and $\mathbf{J}_a$ indicate the Jacobian of the tip and the chosen PSP respectively and $\dot{\mathbf{p}}_{\mathcal{O}_b}$ represents repulsive potential produced by the closest obstacle.

The overall pose estimation and obstacle avoidance algorithm is combined as shown in Figure 3.

Mechanical Constraint Avoidance

Other problem that needs to be addressed for this kind of manipulator is the inherent mechanical limitation which can disturb the movement of the manipulator in avoiding obstacle. In a tendon-driven manipulator, the tendons length l_{ij} can only be in the region of $(1-\zeta)L < l_{ij} < (1+\zeta)L$ where

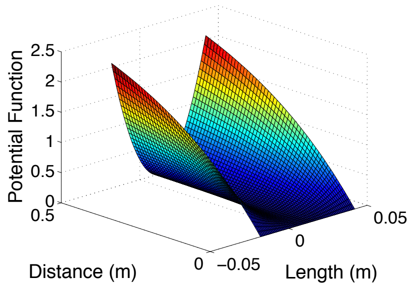


Figure 4: Illustration of the proposed potential function designed to satisfy mechanical constraint as function of the tendon's length and end-effector-to-target Euclidean distance.

ζ and L represent extension ratio and normal length respectively.

We propose an attractive potential in actuator space to attract the length towards normal length L, and thus, makes the tendons avoid mechanical constraint. The potential function is described as

$$U_{lim}(\mathbf{q}) = \sum_{i=1}^{N}\sum_{j=1}^{3}\sigma\left(\frac{l_{ij}-L}{\zeta L}\right)^2, \tag{13}$$

where σ is positive constant. The attractive velocity field is as follows

$$\dot{\mathbf{q}}_{lim}(\mathbf{q}) = -2\sigma\frac{1}{\zeta^2 L^2}(\mathbf{l}-\mathbf{L}), \tag{14}$$

where $\mathbf{l} = [l_{11} \quad \ldots \quad l_{13} \quad l_{21} \quad \ldots \quad l_{N3}]^T$ and $\mathbf{L} = L\mathbf{1}_{3N\times 1}$.

Moreover, we weight this potential field by a weight function w which will reduce the contribution of the mechanical constraint field as the end effector approaches the target position, as follows

$$w(\mathbf{x}) = (1 - e^{-\mu\|(\mathbf{x}-\mathbf{x}_d)\|}), \tag{15}$$

where μ is positive constant.

Figure 4 shows an illustration of this potential as a function of tendons length as well as the distance between end-effector and the target. The final proposed mechanical constraint avoidance in actuator space is given by

$$\dot{\mathbf{q}}_{new}(\mathbf{q},\mathbf{x}) = \begin{bmatrix}\mathbf{0}_{6\times 1}\\ w(\mathbf{x})\dot{\mathbf{q}}_{lim}(\mathbf{q})\end{bmatrix} \tag{16}$$

so that the total potential is given by

$$\dot{\mathbf{q}}(\mathbf{q}) = \mathbf{J}_e(\mathbf{q})^+\dot{\mathbf{x}}_{\mathbf{x}_d}(\mathbf{x}) + \sum_{a=1}^{m}\sum_{b=1}^{n}\mathbf{J}_a(\mathbf{q})^+\dot{\mathbf{x}}_{\mathcal{O}_b}(\mathbf{x}) + \dot{\mathbf{q}}_{new}(\mathbf{q},\mathbf{x}) \tag{17}$$

Results and Discussion

Here, we present the implementation of our approach both in the simulation and the experiment. The combined pose estimation and obstacle avoidance for single segment case is tested on a model of continuum manipulator running in a real-time simulation environment. The obstacle avoidance algorithm without the pose estimation is also validated in an experiment as presented in our publication (Ataka et al. 2016). Moreover, we also applied the obstacle avoidance and mechanical constraint avoidance to a model of multi-segment continuum manipulator as presented in the same publication. A moving obstacle, assumed to be a 5-mm-radius sphere, moves at a constant speed in the surrounding of the manipulator to simulate the part of human body's movement.

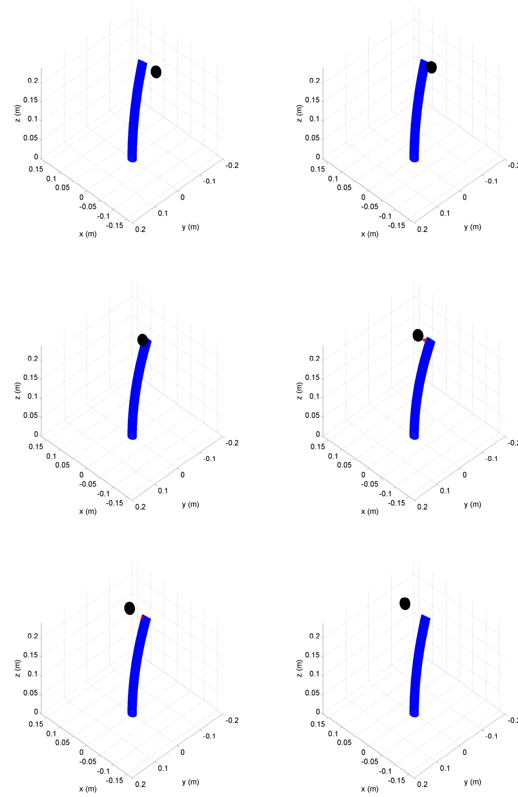
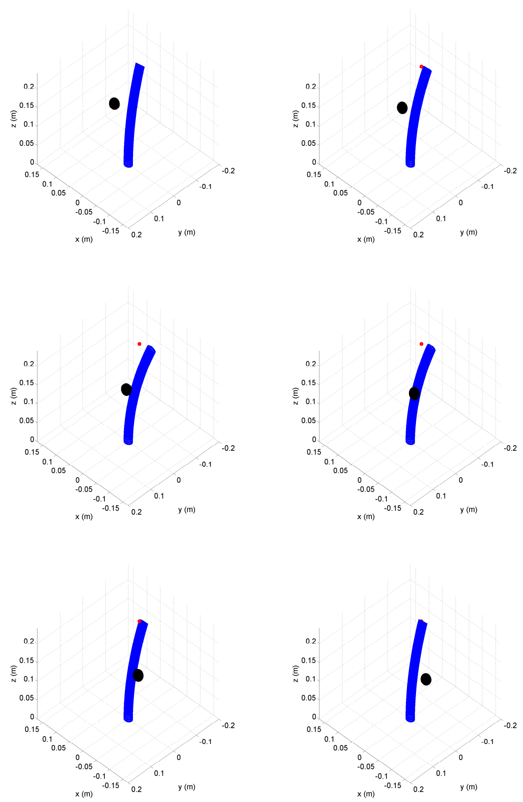


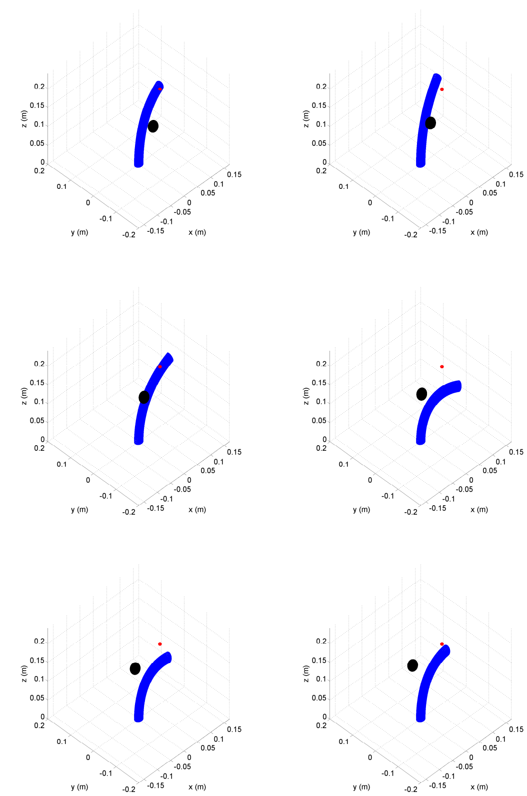
Figure 5: A single segments continuum manipulator's movement with a static target position (small red dot) when obstacle (black sphere) moves close to the tip. The order of movement is as follows: upper left picture, upper right picture, lower left picture, and finally lower right picture.

Single Segment Case

In this subsection, we will first present the simulation results of a combined pose estimation and obstacle avoidance. For the pose estimation, to simulate the sensor, an ideal kinematic model with added Gaussian noise is used. This perfect kinematic model has a true state $\mathbf{x}_k$ updated at every iteration based on the model kinematics. However, the value is assumed to be unaccessible for the obstacle avoidance, which will only capitalize the estimated states $\hat{\mathbf{x}}_k$ from the EKF. The noise has zero-mean and the standard deviation of $\sigma = 10^{-4}$. The variance matrix is then given by $\mathbf{R} = \sigma^2\mathbf{I} \in \mathbb{R}^{3\times 3}$. The black sphere represents the obstacle while the rod dot represents the tip's target, assumed to be



(a)



(b)

Figure 6: A single segments continuum manipulator's movement with a static target position (small red dot) when obstacle (black sphere) moves close to the middle point of the manipulator's body (a) in x-axis direction and (b) y-axis direction. The order of movement for each subfigure is as follows: upper left picture, upper right picture, lower left picture, and finally lower right picture.

fixed.

We show several scenarios, such as an obstacle moves close to the segment's tip, as depicted in Figure 5 and moves close to the middle part of the backbone, as shown in Figure 6a (for the obstacle's movement in x-direction) and Figure 6b (for the obstacle's movement in y-direction). We can conclude that the proposed algorithm works well to improve the safety of the robot's body from collision. The main contribution of the algorithm, however, is demonstrated when the obstacle move at a lower height, such as in Figure 6. Our proposed algorithm enables not only the tip but also the body of manipulator to avoid obstacle.

The obstacle avoidance algorithm, without the pose estimation, has been implemented in an ortho-planar spring tendon-driven continuum manipulator as shown in Figure 1 as presented in (Ataka et al. 2016). The Maxon DC Motors are used as actuators. An electromagnetic-based Aurora sensor coil is embedded in the tip of the manipulator to track the position of the tip. The obstacle is represented by a second sensor coil hung in the air by a thread. The obstacle is assumed to be a sphere with radius of 0.01 m whose centre is specified by the second sensor coil's location. The target is assumed to move in a straight line.

We can see from Figure 7, from left to right, both the 3D view (Figure 7a) and the top view (Figure 7b) of the tip's movement, as well as the comparison between the tip-obstacle distance and the target-obstacle distance (Figure 7c). Without pose estimation, the PSP is assumed to be located only at the tip, hence, only the tip will be safe from collision.

Three Segments Case

Here, we present the performance of our proposed mechanical constraint avoidance in a model of three-segments continuum manipulator. We can see how our proposed algorithm improves the performance of continuum manipulator in tracking the moving target. Without our algorithm, once the tendons length reach their limit, there appears an immediate reduction in the manipulator's maneuverability such that the tip is unable to reach the moving target, as shown in Figure 8a. While, using our algorithm, it is shown in Figure 8b that the manipulator's tip is able to track the moving tar-

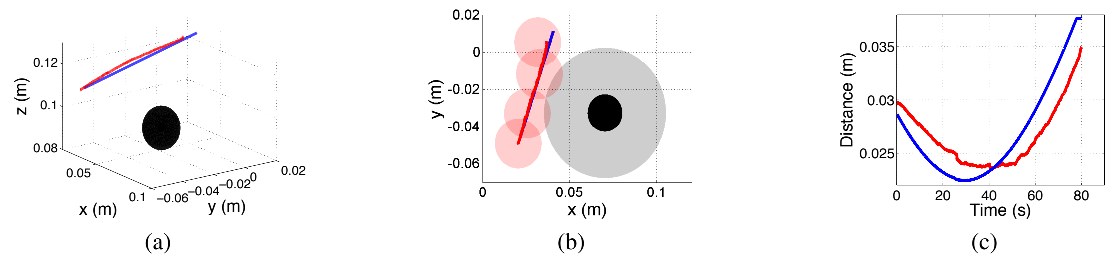


Figure 7: The manipulator's tip (red line) tracks the desired trajectory (blue line) and avoids static obstacle (black spheres) nearby in (a) 3D view and (b) top view. (c) The graph shows the closest distance from the obstacle to the estimated manipulator's body (red line) and to the target (blue line) respectively.

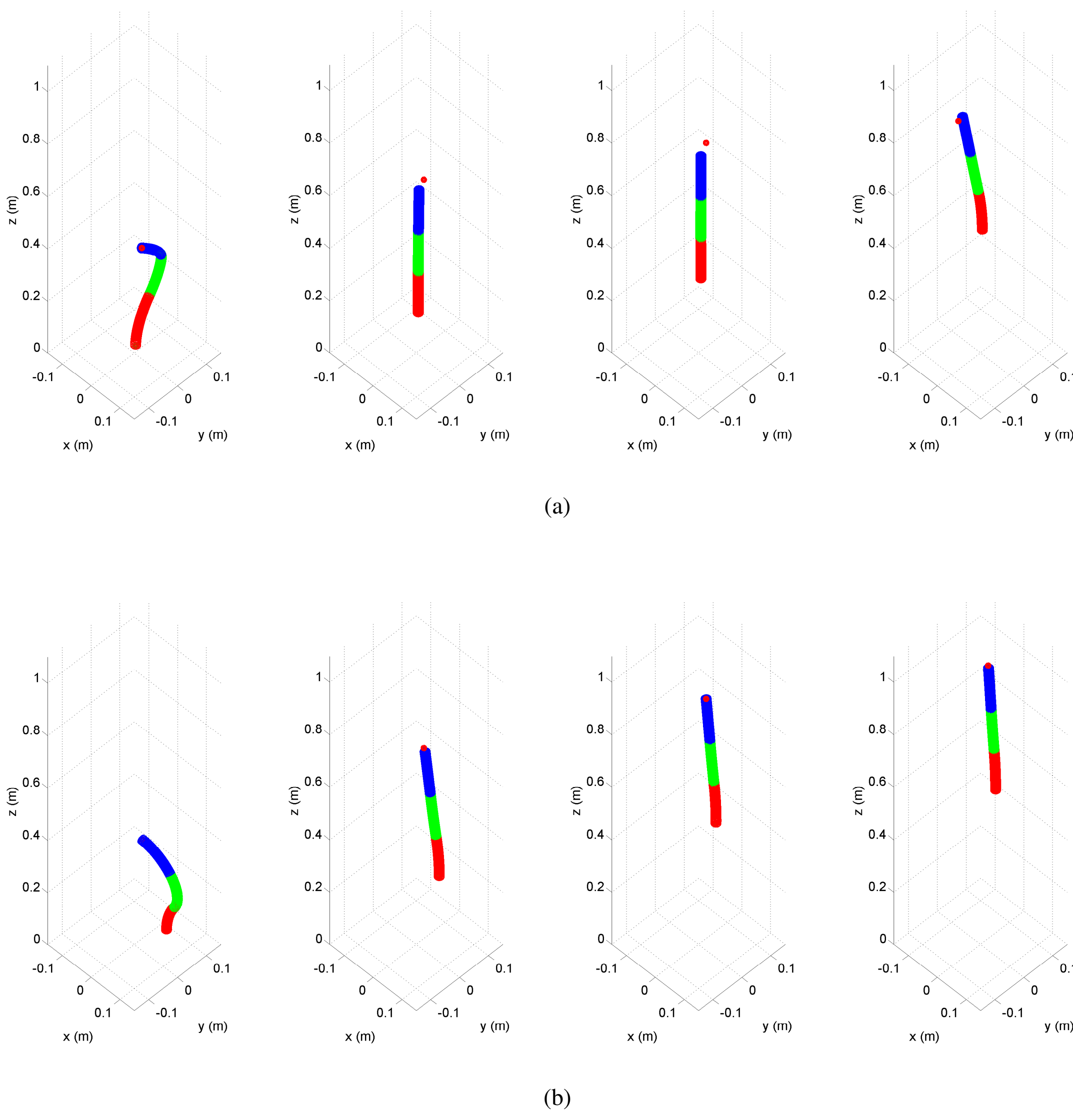


Figure 8: The movement comparison between the continuum manipulator (a) without the proposed algorithm and (b) with the proposed algorithm.

get smoothly. The complete obstacle avoidance simulation results can be seen in (Ataka et al. 2016).

Conclusions and Future Works

In this paper, we present our works on real-time obstacle avoidance, based on modified potential field, for continuum manipulators in dynamic environments. The obstacle avoidance can be equipped with a pose estimator, based on an Extended Kalman Filter, to make the whole body of manipulator safer from collision. The novel potential field in actuator space is also proposed in order to avoid the inherent mechanical constraint of the manipulators. The combined obstacle avoidance and pose estimator is shown to perform well in a simulation for the model of tendon-driven single-segment continuum manipulator. The proposed potential field is also verified in experiment. The extension of the obstacle avoidance for multi-segment case as well as the mechanical constraint avoidance is implemented successfully in a model of three-segments continuum manipulator.

The proposed algorithm has a promising capability to be implemented in a real environment consisting human. For the future works, the whole algorithm can be fully implemented for the real multi-segment continuum manipulators equipped with electromagnetic-based position sensors. The experiment to test the whole algorithm in a continuum manipulator with mobile platform will also be investigated in the future.

Acknowledgement

The work described in this paper is partially supported by the STIFF-FLOP project grant from the European Communities Seventh Framework Programme under grant agreement 287728, the Four By Three grant from the European Framework Programme for Research and Innovation Horizon 2020 under grant agreement no 637095, and the Indonesia Endowment Fund for Education, Ministry of Finance Republic of Indonesia.

References

Ataka, A.; Qi, P.; Liu, H.; and Althoefer, K. 2016. Accepted. Real-Time Planner for Multi-segment continuum manipulator in dynamic environments. *Proceedings of 2016 IEEE International Conference on Robotics and Automation (ICRA)*.

Brock, O., and Khatib, O. 2002. Elastic strips: A framework for motion generation in human environments. *The International Journal of Robotics Research* 21(12):1031–1052.

Burgner-Kahrs, J.; Rucker, D. C.; and Choset, H. 2015. Continuum Robots for Medical Applications: A Survey. *IEEE Transactions on Robotics* 31(6):1261–1280.

Haddadin, S.; Parusel, S.; Belder, R.; and Albu-Schffer, A. 2013. Computer Safety, Reliability, and Security: 32nd International Conference, SAFECOMP 2013, Toulouse, France, September 24-27, 2013. Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg. 202–215.

Hirose, S. 1993. *Biologically inspired robots: snake-like locomotors and manipulators*. Oxford science publications. Oxford University Press.

Khatib, O.; Yokoi, K.; Brock, O.; Chang, K.; and Casal, A. 1999. Robots in human environments. In *Robot Motion and Control, 1999. RoMoCo '99. Proceedings of the First Workshop on*, 213–221.

Khatib, O. 1985. Real-time obstacle avoidance for manipulators and mobile robots. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, 500–505.

Kim, B. S., and Song, J. B. 2010. Hybrid dual actuator unit: A design of a variable stiffness actuator based on an adjustable moment arm mechanism. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 1655–1660.

McMahan, W.; Chitrakaran, V.; Csencsits, M.; Dawson, D.; Walker, I.; Jones, B.; Pritts, M.; Dienno, D.; Grissom, M.; and Rahn, C. 2006. Field trials and testing of the OctArm continuum manipulator. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2336–2341.

Ozgoli, S., and Taghirad, H. 2006. A survey on the control of flexible joint robots. *Asian Journal of Control* 8(4):332–344.

Webster, III, R. J., and Jones, B. A. 2010. Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review. *Int. J. Rob. Res.* 29(13):1661–1683.

Nested Safety Sets for Collision Avoidant Human-Robot Systems

Kelsey P. Hawkins, Henrik I. Christensen

Abstract

We investigate the problem of avoiding collision in fast, dynamic human-robot interaction. Optimal control policies are computed for both the human and robot backwards in time from collision. To represent human unpredictability, the robot models the human using a series of increasingly conservative nondeterministic control models. A collection of nested safety sets is found, each of which provides a degree of safety based on how conservative a model the robot must assume of the human to guarantee safety. This allows the robot to anticipate the safety of an interaction based on states labeled as guaranteed safe, guaranteed unsafe, and levels in-between.

Motivation

In some human-robot interaction scenarios, the human or robot needs to leverage the other's collision avoiding behavior in order to be effective. In order to merge onto a busy highway, a car might need to pull in front of another, recognizing that the other must brake in order to avoid hitting them. While this is less safe than waiting for a clearing, it's not irrational behavior if collision is still reasonably avoidable. On the other hand, if the robot can still be effective while guaranteeing safety, the robot should know where and how this is possible.

If the robot is aware of the risks of an interaction scenario before it begins, the robot can make precursory decisions to improve safety. For example, when approaching a corridor intersection, the robot can slow down before entering to ensure the robot has enough time to react if a human appears around the corner. While the robot cannot estimate precisely how the human will react, it can anticipate worst-case behavior and prepare for everything in-between. By having explicit representations of safe regions and its assumptions about the human, the robot can responsibly leverage risk.

Methods

Joint Dynamics Model

The joint human-robot dynamics are modeled as $\dot{x}_{\mathrm{HR}} = f(x_{\mathrm{HR}}, w_{\mathrm{HR}})$ where $x_{\mathrm{HR}} \in \mathcal{X}_{\mathrm{HR}}$ encodes both the human and robot's state and $w_{\mathrm{HR}} = (w_{\mathrm{H}}, w_{\mathrm{R}}) \in \mathcal{W}_{\mathrm{H}} \times \mathcal{W}_{\mathrm{R}}$ is joint control input provided by the human and robot respectively. The sets $\mathcal{W}_{\mathrm{H}}$ and $\mathcal{W}_{\mathrm{R}}$ represent what is believed to be the physical limits of the agents, thus providing a limit to what

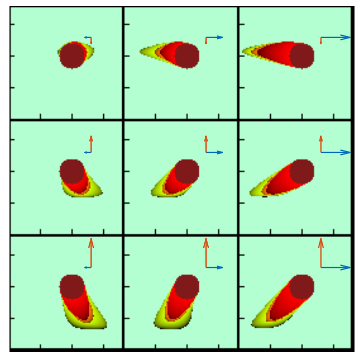


Figure 1: Nested safety sets for different initial velocity conditions in the state space, magnitudes indicated in the top right corner of each square. Each grid element labels the safety of a trajectory with those initial positions for the robot and human. Green elements can be guaranteed not to collide by the robot. Red elements are guaranteed to collide, regardless of human or robot behavior. Orange elements can only escape collision with near optimal control by the human and robot. Yellow elements are a short time away from collision.

is dynamically possible and impossible. The *collision set* $\mathcal{C} \subset \mathcal{X}_{\mathrm{HR}}$ is the closed set of states where the human and robot are considered instantaneously in collision.

Say we are given a model of human control $\mathcal{U}_{\mathrm{H}} = (U_{\mathrm{H}}, V_{\mathrm{H}})$ where U_{H} is the extent to which the human optimizes their input to avoid collision, and V_{H} is a set of human uncertainty tolerated by the robot. For a given state, the robot will predict an optimal control taken by the human, $u_{\mathrm{H}}^* \in U_{\mathrm{H}}$ and will consider human input to follow the model if it is near optimal, that is, it falls within the set $w_{\mathrm{H}} \in u_{\mathrm{H}}^* + V_{\mathrm{H}} \subseteq \mathcal{W}_{\mathrm{H}}$.

We can build a hierarchy of increasingly conservative control models to assess how much variation the robot is accounting for. The least conservative represents the human acting perfectly optimally, $\mathcal{U}_{\mathrm{H}}^{\mathrm{LC}} = (\mathcal{W}_{\mathrm{H}}, \{0\})$. We can relax this assumption slightly if we assume constant variance bounded by $V_{\mathrm{H}}^{\mathrm{Var}}$ and that the control is optimized over the remainder of the input domain $U_{\mathrm{H}}^{\mathrm{Var}} = \mathcal{W}_{\mathrm{H}} \dot{-} V_{\mathrm{H}}^{\mathrm{Var}}$. Finally, the most conservative model assumes all physically capable inputs may be executed by the human $\mathcal{U}_{\mathrm{H}}^{\mathrm{MC}} = (\{0\}, \mathcal{W}_{\mathrm{H}})$.

Semi-Cooperative Differential Game

When the human and robot are near collision, we assume that they play a differential game where value is determined by the minimum time functional

$$\omega_{\mathcal{M}}(x(\cdot)) := \inf\{t \geq 0 : x(t) \in \mathcal{M}\}, \qquad (1)$$

returning the first time an evolution hits an unsafe set $\mathcal{M}$. Consider the value function

$$\omega_{\mathcal{M}}[\mathcal{U}](x_0) := \sup_{u_{\mathrm{HR}}(\cdot)} \inf_{v(\cdot)} \omega_{\mathcal{M}}(x[x_0, u_{\mathrm{HR}}(\cdot) + v_{\mathrm{H}}(\cdot)]). \quad (2)$$

This encodes the idea that along trajectories, both the robot and the optimal component of the human's input are trying to increase the minimum time to collision and that they are both opposed by the suboptimal component of the human's input. Under the least conservative model, this game will be fully cooperative, and under the most conservative it becomes a classic differential game of human v.s. robot.

Solving for this function can assign a value to every state in the state space. In general, higher values are safer because collision is guaranteed to be avoided for a longer period of time. If we can assume that collision is still avoidable, the human has more time to react and choose a safer course of action.

If the value is $\omega_{\mathcal{M}}[\mathcal{U}](x_0) = \infty$, we can assume this control model to be safe for all time for that initial starting state and control model. Accumulate these initial states into the safety set, $\mathrm{Safe}[\mathcal{U}, \mathcal{M}]$. These sets are computed using a level set method.

Nested Safety Sets

More conservative models shrink the size of $\mathrm{Safe}[\mathcal{U}, \mathcal{M}]$ but provide a higher safety guarantee. Suppose for a more conservative model $\mathcal{U}$, the unsafe set is $\mathcal{M} = \mathrm{Safe}[\mathcal{U}', \mathcal{M}']^{\mathrm{c}}$ for some less conservative model $\mathcal{U}'$. Continue this recursive nesting until you have the least conservative model and the unsafe set is the collision set $\mathcal{M} = \mathcal{C}$. Then, we can form a preference relation from this hierarchy of safe initial conditions based on what the highest level safety set the initial state is in. Further, the minimum time function can give a fine-grained valuation for how soon in time that state is to falling into the weaker safety set, based on the worst case human input for that model.

Application: Tight Corridor Intersection

Suppose a human and robot are moving down intersecting corridors as visualized in Fig. 2. We can encode the joint state as $x_{\mathrm{HR}} = (p_{\mathrm{H}}, v_{\mathrm{H}}, p_{\mathrm{R}}, v_{\mathrm{R}})$ where p is position and $\dot{p} = v$ is velocity. We assume proportional velocity control for both the human and robot, $\dot{v} = k(u - v)$, where u is a target velocity, the control value for each agent.

We assume that, near collision, the robot is acting optimally but uses three control models for the human in computation of the nested safety sets. These sets are visualized in Fig. 1. The model $\mathcal{U}_{\mathrm{H}}^{\mathrm{LC}}$ yields the safe set which excludes only the light and dark red states. Next, the model $\mathcal{U}_{\mathrm{H}}^{\mathrm{Var}}$ removes orange states from the previous set. Finally, the model $\mathcal{U}_{\mathrm{H}}^{\mathrm{MC}}$ removes yellow states from the previous set.

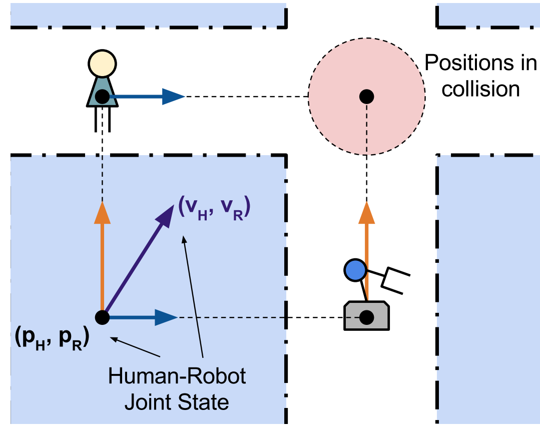


Figure 2: The tight corridor intersection problem, encoded as a joint human-robot system.

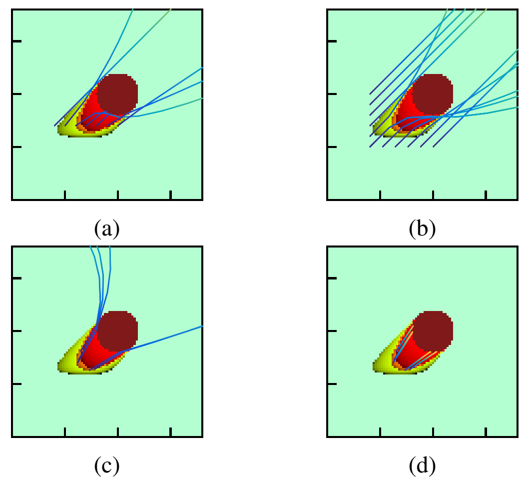


Figure 3: Interaction trajectories under various models and initial positions.

Thus, starting in pale green states, the system can insure safety, regardless of the human's control. Darker values represent higher values of $\omega(x_0)$ for the relevant control model.

Fig. 3 demonstrates how a simulated human-robot interaction behaves under different initial positions for equivalent initial velocities. Unless constrained by their model, the human and robot try to control to their constant initial velocities. In fig. 3a, the human chooses optimal controls in the control model of the safety set just exited. It shows that initial positions in the red sets are guaranteed to end in collision, that orange states will just barely miss with evasive action, and that green positions require little course alteration. Fig. 3b shows that starting in green, collision can be avoided with even least-conservative $\mathcal{U}_{\mathrm{H}}^{\mathrm{LC}}$ behavior of the human. Fig. 3c and shows that starting in orange, collision can be avoided only if the human is acting near optimally, while Fig. 3d shows that, for the same initial states, if the human is acting just short of optimal then collision will occur.

Dynamic Task Planning for Safe Human Robot Collaboration

Giulio Bernardi, Amedeo Cesta, Andrea Orlandini
CNR – National Research Council of Italy
Institute for Cognitive Science and Technology
{name.surname}@istc.cnr.it

Alessandro Umbrico
Roma TRE University
Department of Engineering
alessandro.umbrico@uniroma3.it

Abstract

The collaboration between humans and robot is a current technological trend that faces various challenges, among these the seamless integration of the respective work capabilities. Industrial robots have demonstrated their capacity to meet the needs of many applications, offering accuracy and efficiency, while humans have both experience and the capability to elaborate over such experience that are absolutely not replaceable at any time. Clearly the symbiotic integration of humans and robots opens to some new problems: the effective collaboration requires intelligent coordination. This paper present an example of interactive environment for facilitating the collaboration between humans and robot in performing a shared task. In particular we describe a tool based on AI planning technology to help the smooth intertwining of activities of the two actor in the work environment. The paper presents a case study from a real work environment, describes a comprehensive architectural approach to the problem of coordinated interaction, and then presents details on the current status of the approach.

Introduction

Industrial robots have demonstrated their capacity to meet the needs of many applications, offering accuracy and efficiency. However, when robot-worker collaboration is needed, there are a number of open issues to be taken into account, first of those is human safety that needs to be enforced in a comprehensive way. In this regard, seamless and safe human-robot collaboration still constitutes an open challenge in manufacturing. These authors are currently working in the FourByThree research project[1] aimed to design, build and test pioneering robotic solutions able to collaborate safely and efficiently with human operators in industrial manufacturing companies. Its overall aim is to respond to the above challenge by creating a new generation of robotic solutions, based on innovative hardware and software, which present four main characteristics: modularity, safety, usability and efficiency And considers three different actors: humans, robots and the environment. The resulting robotic solutions of the project will be tested in four pilot implementations, which correspond to real industrial needs and are representative of the two possible robot-human relationships in a given workplace without physical fences: coexistence (human and robot conduct independent activities) and collaboration (they work collaboratively to achieve a given goal).

A key feature in FourByThree is the pursuing of both hardware development, the synthesis of an innovative robotic arm, and software development, the design and realization of of a software environment integrating several new ideas to guarantee human safety during operations. The planning based environment described in this paper is one of such innovative software modules.

This work shows the general design of an environment for facilitating both the specification of a subdivision of roles between robot and human in performing a task, then the temporal planning of a flow of actions that guarantee safe coordination, then an initial description of a run time support for the symbiotic execution of such a multi-agent plan.

Human-Robot Collaborative Scenarios

A human-robot collaboration workcell can be considered as a bounded connected space with two agents located in it, a human and a robot system, and their associated equipment (Marvel, Falco, and Marstio 2015). A robot system in a workcell consists of a robotic arm with its tools, its base and possibly additional support equipment. The workcell also includes the workpieces and any other tool associated with the task and dedicated safeguards (physical barriers and sensors such as, e.g., monitoring video cameras) in the workcell space. In such workcell, different degrees of interaction between a human operator and the robot can be considered (Helms, Schraft, and Hägele 2002). In all these cases, it is assumed that the robot and the human may need to occupy the same spatial location and interact according to different modalities: *Independent*, the human and the robot operate on separate workpieces without collaboration, i.e., independently from each other. *Synchronous*, the human and the robot operate on sequential components of the same workpiece, i.e., one can start a task only after the other has completed a preceding task. *Simultaneous*, the human and the robot operate on separate tasks on the same workpieces at the same time. *Supportive*, the human and the robot work cooperatively in order to complete the processing of a single workpiece, i.e., they work simultaneously on the same task.

[1] http://www.fourbythree.eu

Different interaction modalities requires the robot endowed with different safety (hardware and control) settings while executing tasks.

A Case Study: ALFA Pilot

In the FourByThree project, four different pilots are considered covering different production processes, i.e., assembly/disassembly of parts, welding operations, large parts management and machine tending. Among these, here we describe the ALFA Pilot as a specific case study for the paper. This case study corresponds to a production industry (the *ALFA PRECISION CASTING*[2]) which represents a real working scenario with different relevant features for our objectives (e.g. space sharing, collaboration; interaction needs, etc).

The overall production process consists of a metal die which is used to produce a wax pattern in a injection machine. Once injected, the pattern is taken out the die. Several patterns are assembled to create a cluster. The wax assembly is covered with a refractory element, creating a shell (this process is called investing). The wax pattern material is removed by the thermal or chemical means. The mould is heated to a high temperature to eliminate any residual wax and to induce chemical and physical changes in the refractory cover. The metal is poured into the refractory mould. Once the mould has cooled down sufficiently, the refractory material is removed by impact, vibration, and high pressure water-blasting or chemical dissolution. The casting are then cut and separated from the runner system. Other post-casting operations (e.g. heat treatment, surface treatment or coating, hipping) can be carried out, according to customer demands.

In this paper we focus on the first step (preparation of the die for wax injection and extraction of the pattern from the die) which is a labour demanding operation that has a big impact on the final cost of the product. Specifically, the operation consists of the following steps: (i) mount the die; (ii) inject the wax; (iii) open the die and remove the wax; (iv) repeat the cycle for new patterns.

The most critical sub-operation is the opening of the die because it has a big impact on the quality of the pattern. In this context, the involvement of a collaborative robot has been envisaged to help the operator in the *assembly/disassembly* operation.

Assembly/disassembly operation. Due to the small size of the dies and the type of operations done by the worker to remove the metallic parts of the die, it is very complex for the robot and the worker to operate on the die simultaneously. However, both of them can cooperate in the assembly/disassembly operation.

Once the injection process has finished, the die is taken to the workbench by the worker. The robot and the worker unscrew the bolts of the top cover, see Figure 1. There are nine bolts, the robot starts removing those closer to it, and the worker the rest. The robot unscrews the bolts on the cover by means of a pneumatic screwdriver. The worker

Figure 1: Cover removal

removes the top cover and leaves it on the assembly area (a virtual zone that will be used for the re-assembly of the die). The worker turns the die to remove the bottom die cover. The robot unscrews the bolts on the bottom cover by means of a pneumatic screwdriver. Meanwhile the operator unscrews and remove the threaded pins from the two lateral sides to release the inserts. The worker starts removing the metallic inserts from the die and leaves them on the table, see Figure 2. Meanwhile, the robot tightens the parts to be

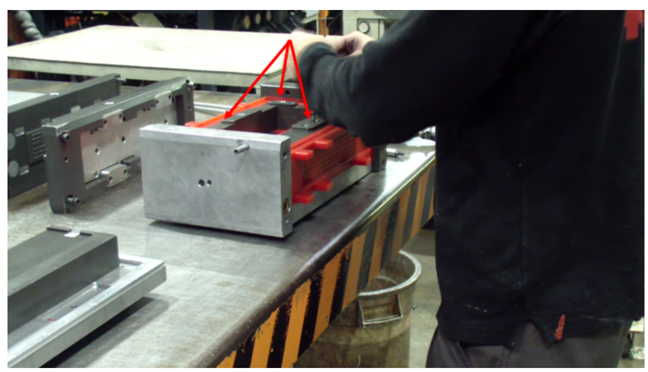


Figure 2: Insert removal

assembled/reassembled together screwing bolts. The worker re-builds the die. The worker and the robot screw the closing covers.

Thus the human and the robot must collaborate to perform assembly/disassembly on the same die by suitably handling different parts of the die and screwing/unscrewing bolts. Specifically, the human worker has the role of leader of the process while the robot has the role of subordinate with some autonomy. Moreover, the robot must be able to manage a screwdriver device and monitor the human location and its activities.

[2] A medium sized company producing aluminium parts for different industries for applications that are characterized by low size production batches and requiring tight tolerance and dimensional precision.

A Dynamic Task Planning Framework for Safe Human-Robot Collaboration

As part of the overall FourByThree control architecture, a dynamic task planner is to provide continuous task synthesis features, safety critical properties at execution time, and user modelling ability for adapting tasks to the particular human at work. The integration of plan synthesis and continuous plan execution has been demonstrated both for timeline based planning (e.g., (Py, Rajan, and McGann 2010)) and PDDL based (e.g., (Cashmore et al. 2014)). In scenarios of human robot interaction important problems have been addressed: (a) ”human aware” planning has been explored for example in (Sisbot and Alami 2012), (b) the interaction of background knowledge for robotic planning in rich domain (addressed for example in (Lemaignan and Alami 2013), (c) synthesis of safety critical plans to guarantee against harmful states (relevant in co-presence with humans) is addressed in (Abdellatif et al. 2012) and (Orlandini et al. 2013)). Within the FourByThree project, a timeline-based planning approach is pursued relying on the APSI-TRF software infrastructure (Cesta and Fratini 2008), made available by European Space Agency, and improved from the initial proposal (Cesta et al. 2009) and its test in several missions. The overall framework is depicted in Figure 3. A Production Engineer is in charge of defining the Human-Robot collaborative (HRC) production process characterizing each task according to specific HRC settings (i.e., interaction modalities). In the ALFA pilot, the production engineer has been provided with a suitable interface to define the human-robot collaborative process and setting the proper interaction modalities. Figure **??** shows a view of such interface.

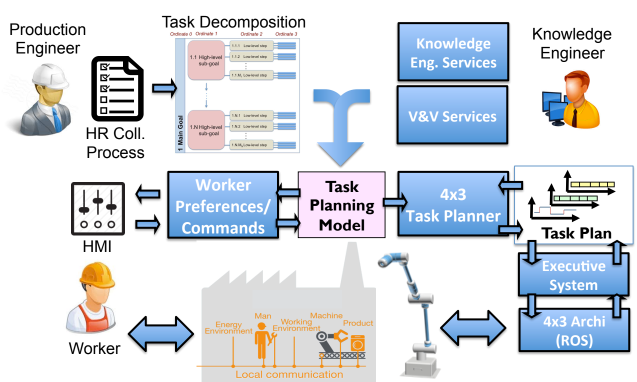


Figure 3: Production Engineer Interface showing the Assembly process settings.

Then, a Knowledge Engineer is to encode such information in a task planning model following a hierarchical decomposition (Stanton 2006) and leveraging the features provided by an environment for Knowledge Engineering of Planning with Timelines, called KEEN (Orlandini et al. 2014), that integrates “classical” knowledge engineering features with Validation and Verification (V&V) formal techniques to perform domain model validation, planner validation, plan verification, etc. The integration of Planning and Scheduling (P&S) technology with V&V techniques is key to synthesize a safety critical controller for the robot acting in a symbiotic environment with a human worker as partner. Then, the Task Planning Model can be adapted also according to the preferences of the Human Worker that is supposed to interact with the robot during the production process. A FourByThree Task Planner generates a temporally flexible task plan to be dispatched to the robot through an Executive System integrated in the overall FourByThree (ROS-based) architecture. During the production process, the Executive System is also in charge of monitoring the plan execution and, in case of need (e.g., a specific command issued by the human worker), asks the task planner to dynamically adapt the control strategy to face the modifications in the production environment.

Timeline-based Planning

The dynamic task planning framework introduced above has been designed and implemented pursuing a Timeline-based approach to planning. This planning approach has been introduced in early 90s (see for instance (Muscettola 1994)) and takes inspiration from the classical control theory. It models a complex system by identifying a set of relevant features that must be controlled over time. A timeline-based application aims at controlling a complex system by synthesizing temporal behaviors of its features (i.e. *timelines*).

There are several timeline-based systems that have been introduced in the literature (Barreiro et al. 2012; Chien et al. 2010; Fratini et al. 2011; Laborie and Ghallab 1995) and also several attempts of formalizations have been made, see for example (Cimatti, Micheli, and Roveri 2013). The reader may refer to the work (Cialdea Mayer, Orlandini, and Umbrico 2015) which represents a complete and comprehensive formalization of timeline-based planning concepts. In this section we provide a brief and short description of timeline-based approach by following the formalization in (Cialdea Mayer, Orlandini, and Umbrico 2015) which takes into account also controllability properties of planning domains and plans. Indeed, not every action in a plan is under the system control, as events exist that depend on the *environment*. The execution of a plan is then usually under the responsibility of an executive system that forces system transitions dispatching commands to the concrete system, while continuously accepting feedback and, thus, monitoring plan execution. In such cases, the execution time of controllable tasks should be chosen so that they can face uncontrollable events. This is known as the *controllability problem* (Vidal and Fargier 1999).

Timeline-based planning domains. A timeline-based planning domain is modeled as a set of features that must be controlled over time. These features are modeled by means of *multi-valued state variables* that specify causal and temporal constraints characterizing the allowed temporal behaviors of the domain features. Namely, state variables model domain features behaviors by not considering specific application environment or the behaviors of other features of the domain. Formally, a state variable x is described by the tuple (V, T, γ, D). V is the set of values $v \in V$ the variable x may assume over time. A value represents either an action or a state the related domain feature may perform or assume respectively. The *state variable transition function* $T : V \rightarrow 2^V$ describes for each value $v_i \in V$ the set of values $v_j \in V$ (where $i \neq j$) that are allowed to follow v_i. The *controllability tagging function* $\gamma : V \rightarrow \{c, u\}$ provides information about the controllability properties of state variable's values. If a value $v \in V$ is tagged as *controllable* ($\gamma(v) = c$) then the system (e.g. the planner or the executor) can decide the actual duration of the value. If a value $v \in V$ is tagged as *uncontrollable*, instead, ($\gamma(v) = u$) then the system cannot control the duration of the value, the value is under the control of the *environment*. The *state variable duration function* $D : V \rightarrow R \times R$ provides duration constraints by setting upper and lower bounds to the duration of each value $v \in V$.

The behavior of state variables may be further restricted by means of *synchronization rules*. Synchronization rules allow to specify temporal constraints between values of different state variables in order to coordinate their (temporal) behaviors. Namely, while state variables specify *local* rules for the single features of the domain, synchronizations represent *global* rules specifying how different features of the domain must behave together. A formal definition of a synchronization rule is the following:

$$a_0[x_0 = v_0] \rightarrow \exists\, a_1[x_1 = v_1]...a_n[x_n = v_n].\, C$$

where (i) $a_0, ..., a_n$ are distinct *token variables* (i.e. variables denoting valued temporal intervals of state variables); (ii) for all $i = 0, ..., n$, x_i is a state variable and v_i is a value of x_i; and (iii) C is a *positive boolean formulae* (PBF) specifying temporal constraints among token variables and where only the token variables $a_0, ..., a_n$ occur. The left-hand part of the synchronization $a_0[x_0 = v_0]$, is called the *trigger* of the rule. Intuitively a synchronization rule of the above form requires that, whenever the state variable x_0 assumes the value v_0 in some interval a_0, there are tokens a_i ($1 \leq i \leq n$) where the variable x_i has the value v_i and one of the possible choices for making C true holds.

Thus a *planning domain* is composed by a set of state variables and a set of synchronization rules. Specifically, we distinguish two types of state variables. State variables, called *planned variables* that model the features of the domain the system can control (or partially control) and state variables, called *external variables*, that model features of the domain concerning the environment and that the system cannot control. External state variables allow to model features of the *environment* the system cannot control but that must care about in order to successfully carry out activities.

Flexible timelines and plans. Planning with timelines usually entails considering sequence of valued intervals and time flexibility is taken into account by requiring that the durations of valued intervals, called *tokens*, range within given bounds. In this regard, a flexible plan represents a whole set of non-flexible timelines, whose tokens respect the (flexible) duration constraints. Formally a token is defined as a triple of the form $(v, [e, e'], [d, d'])$ where $v \in V$ is a value of a state variable $x = (V, T, \gamma, D)$, $e, e', d, d' \in R$, $e \leq e'$ and if $D(v) = (d_{min}, d_{max})$, then $d_{min} \leq d \leq d' \leq d_{max}$. If $\gamma(v) = c$, then the token is *controllable* and if $\gamma(v) = u$, then the token is *uncontrollable*. If x is an external variable, then any token for x is uncontrollable and it is called *observed token*.

Thus, a *timeline* for a state variable $x = (V, T, \gamma, D)$ in the temporal horizon H is finite sequence of tokens for x:

$$FTL_x = x^1 = (v_1, [e_1, e'_1], [d_1, d'_1]), ..., x^k = (v_n, [e_n, e'_n], [d_n, d'_n])$$

where the sequence of values $v_1, ..., v_n$ of the tokens satisfy the transition constraints of the state variable.

Note that a flexible timeline represents an envelop of possible behaviors. We refer to the behaviors of the timelines as the *schedules* of the timelines. A schedule of a timeline is obtained by choosing the exact end time for the tokens of the timeline (i.e. by scheduling its tokens). In particular, a schedule of a timeline is a sequence of scheduled tokens that satisfy the duration constraints of the related values.

A set of flexible timelines do not convey enough information to represent a flexible plan. Indeed, the representation fo a flexible plan must include also information about the relations that have to hold between tokens in order to satisfy the synchronization rules of the planning domain. Thus a flexible plan Π over the horizon H is formally defined as the pair (FTL, R), where FTL is a set of flexible timelines and R is a set of relations on tokens representing a possible choice to satisfy synchronization rules S of the domain.

Timeline-based problems and solutions. Given the concepts described above, a *planning problem* can be formally defined as the tuple (D, G, O, H) where: (i) $D = (P, E, S)$ is a planning domain; (ii) G is a planning goal which specifies a set of token variables and constraints to satisfy; (iii) $O = (FTL_E, R_E)$ is the *observation* which describes a set of flexible (FTL_E) timelines for all the external variables of the domain and R_E is a set of temporal constraints that hold between the tokens of FTL_E; (iv) H is the *temporal horizon*.

A flexible plan Π is a solution for a planning problem if it satisfies the planning goal and if it does not make any hypothesis on the behaviors of the external variables (i.e. if the plan does not change the *observation* of the problem). Formally, given a planning problem $P = (D, G, O, H)$ and a flexible plan $\Pi = (FTL, R)$ over the temporal horizon H', Π is a flexible solution for P if: (i) $H = H'$; (ii) Π is valid w.r.t. the domain; (iii) Π satisfies the planning goal G; (iv) $FTL_E \subseteq FTL$ where $O = (FTL_E, R_E)$.

EPSL - Extensible Planning and Scheduling Library. The Extensible Planning and Scheduling Library (EPSL) is a P&S tool that has been developed to solve planning problems by exploiting the semantics of flexible timelines provided in (Cialdea Mayer, Orlandini, and Umbrico 2015) and summarized above. It is the result of a research effort started after the analysis of different timeline-based systems, such as (Fratini, Pecora, and Cesta 2008; Barreiro et al. 2012;

Laborie and Ghallab 1995), as well as some previous experiences in deploying timeline-based solvers for real world problems (Cesta et al. 2011; Fratini et al. 2011). EPSL relies on the APSI (Cesta and Fratini 2008) modeling framework which provides a timeline-based representation framework for planning domains and problems. On top of such functionalities EPSL provides a software machinery which allows users to easily define timeline-based planners by specifying strategies and heuristics to be applied in specific applications. A more detailed description of the system architecture is provided in (Umbrico, Orlandini, and Cialdea Mayer 2015). However, broadly speaking the EPSL key point is the *planner interpretation*. According to this interpretation, a timeline-based planner is defined as the composition of several solving procedures. A planner is defined as a tuple (S, H, E) where: (i) S is the strategy used to manage the fringe of the search space (the framework provides several built-in options like A^*, DFS, BFS, etc); (ii) H is the heuristic function utilized to analyze the current plan by looking for flaws and choosing the most *promising* flaw to solve (currently we apply a *hiearchy-based* heuristic which allows for hierarchical plan decomposition); (ii) E is the resolver engine which encapsulates the set of algorithms that allow to detect and solve flaws of the plan. So an EPSL-based planner follows a classic plan-refinement solving procedure whose behavior and performance can be adapted by changing parameters described.

EPSL provides and enhanced framework for developing applications in which designers may focus on single aspects of the solving process (i.e. a particular heuristic function or a new resolver in order to add reasoning capabilities to the framework) without taking care of all the details related to timeline-based planner implementation.

Dynamic Task Planning

The human-robot collaborative scenarios envisaged in Section consist of a work-cell where a human operator and a robot have a tight collaboration to perform factory operations. The environment is composed of two *autonomous agents* that can perform some *low-level tasks* and that may collaborate in order to carry out operations.

In such a context there are several features the planning framework must care about in order to control the robot and guarantee a safe collaboration with the human operator. Specifically, we can identify three important features to address: (i) **supervision**, to represent and satisfy the production requirements needed to complete the factory processes; (ii) **coordination**, to represent the activities the human operator and the robot must perform according to the Human-Robot Collaboration settings; (iii) **uncertainty**, to manage the *temporal uncertainty* about the activities of the human operator that the system cannot control.

It is worth observing that in such a context the key enabling feature of the dynamic task planning framework is the capability to model and manage the temporal uncertainty about the activities of the human operator. Indeed, the human is not under the control of the system but the robot must care about the human and execute its operations accordingly.

Thus, w.r.t. the timeline-based approach we apply to develop the task planning framework, the human is modeled as a *planned variable* where all values (i.e. the low level tasks the operator may perform) are *uncontrollable*. It means that the dynamic task planning framework can plan for the low-level tasks of the human and the robot to coordinate them. However human activities are are *uncontrollable*, so the system must carry out robot's tasks by monitoring the human's tasks. In this way, the dynamic task planning framework we are developing realizes a *human aware planning mechanism* which provides the robot with the capability to interact with the operator and also to dynamically adapt the plan if some changes occur in the operator's plan. In this sense, we have extended our hierarchical modeling approach described in (Borgo et al. 2016) by introducing supervision and coordination issues.

In the following, we presents the deployment of the envisaged planning framework in the ALFA pilot, introduced in Section . Nevertheless, it is worth observing that our approach can be easily adapted to different pilots following the same modeling approach. Indeed, the deployed planning technology is domain independent while the domain specific information are enclosed in the task planning model.

The hierarchical structure of the timeline-based planning model for the assembly production process implemented in the ALFA pilot is depicted in Figure 4[3]. The supervision

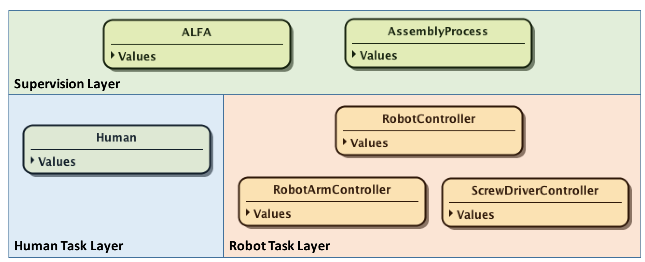


Figure 4: The hierarchy of the task planning domain.

layer represents the elements describing the processes of the work-cell. The *ALFA* state variable in Fig. 4 represents the general ALFA pilot production process and the related operations in which each value represents a specific operation (e.g. the *Assembly operation*). Then the *AssemblyProcess* state variable describes the high-level tasks needed to successfully carry out the *Assembly* process.

In this regard, Fig. 5 describes the first set of requirements that the production engineer has set to properly complete the process. Namely, the model must represent the requirements (i.e. temporal and causal constraints) needed to complete the related factory process correctly. Given, the *Assembly* process of the case study, Fig. 5 shows the sequence of the *high-level tasks* required to properly perform the operation, i.e., the values of AssemblyProcess . It is worth observing that the model is not considering coordination features

[3]The complete planning domain defined for the ALFA pilot is available at the following link: `https://db.tt/rfQDFz1p` with DDL planning syntax.

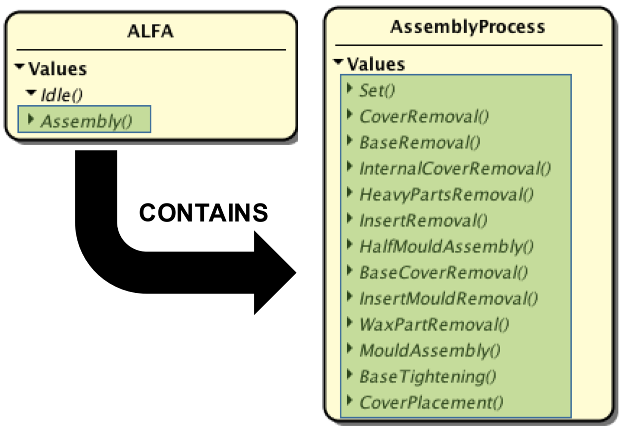


Figure 5: Defining the workflow of work-cell operations.

at this abstraction level. Moreover, each high-level task is composed by a sequence of low-level tasks that can be directly performed by either the human operator or the robot, according to collaborative process defined by the production engineer (including also the type of collaboration).

Thus, the next step is to define the *low-level tasks* required to complete the processes's high-level tasks, assign them to the human or the robot, and select the desired collaboration type. The Fig. 6 shows an example of coordination between the robot and the human to perform the high-level task named *BaseRemoval* of the *Assembly* process. The model describes the sequence of low-level tasks needed to successfully complete the *BaseRemoval* task, and their assignment. In this specific case, the robot supports the human by unscrewing some bolts of the base with selected collaboration type *simultaneous* (the human and the robot work on the same die cover while unscrewing different bolts).

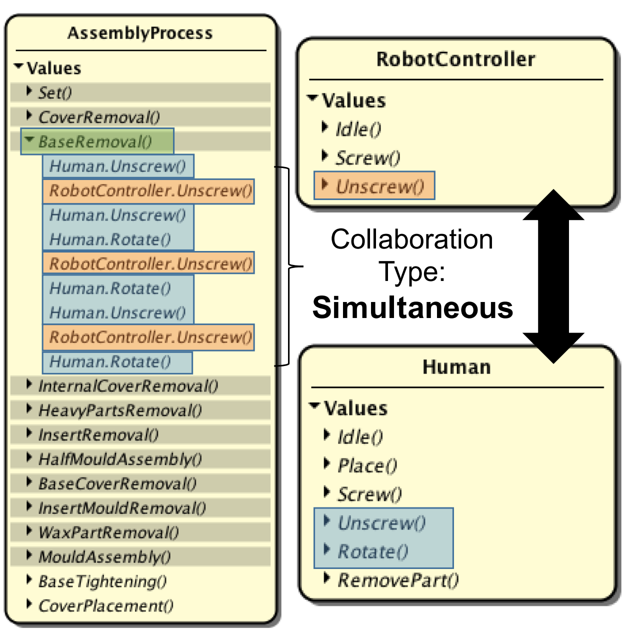


Figure 6: Assigning tasks to the robot and the human operator.

Again, the control system (and the robot) must be *aware of the human* and adapt its tasks according to the human-robot collaboration process defined by the production engineer and human worker preferences/status. The robot system must carry out its activities by monitoring the activities of the human operator. Indeed, the system cannot *decide* the exact duration of the activities of the human which is outside the control of the control system. Thus, the flexible task planning framework must properly manage the uncertainty about the human's behavior in order to properly complete the tasks, even if unexpected events occur (e.g. delays in human activities execution).

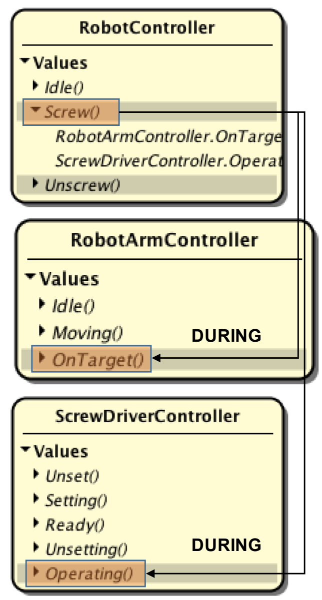


Figure 7: Implementing the low-level commands for the robot's controller.

Finally, the low-level tasks of the robot must be further decomposed in order to synthesize the set of *commands/signals* to be dispatched for execution. Fig. 7 describes the implementation of the low-level task *Screw* of the robot in terms of the position of the robotic arm and the status of the tool used. In Fig. 7 values in boxes and arrows shows a synchronization rule to exemplify a requirement specifying two temporal constraints that requires that the robot sets its arm on a bolt (i.e., *onTarget* value) and must activate the *ScrewDriver tool* (i.e. the *Operative* value) while maintaining the position of the arm.

Figure 8 shows an excerpt of a hierarchical timeline-based plan for the *Assembly* process of the ALFA case study. The horizontal sections (i.e., bars with different colors) partition the plan according to the hierarchy depicted in Fig. 4. The vertical section (in red) depicts an example of high-level task decomposition and application of the synchronization rules of the domain. Namely, the decomposition of the *BaseRemoval* high-level task of the *Assembly* process: the BaseRemoval task requires the human operator and the robot to unscrew some bolts from two lateral side of the work-piece

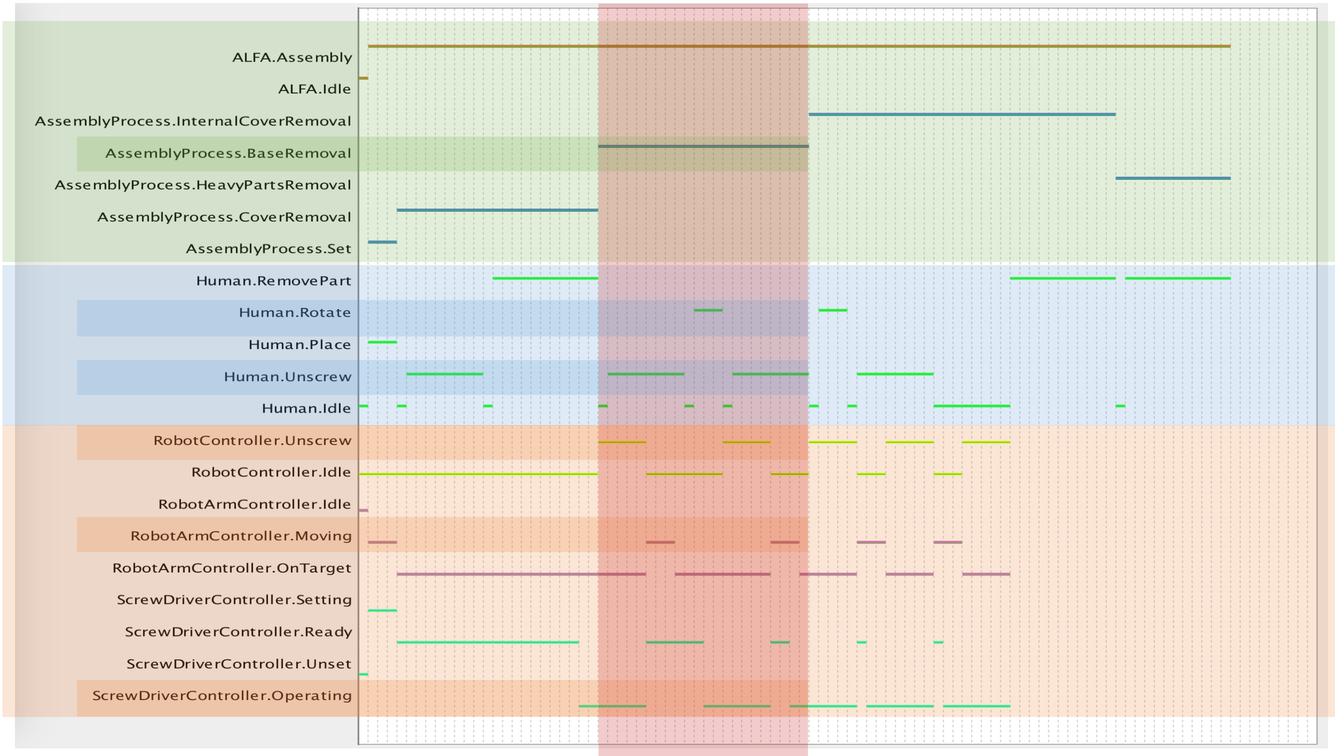


Figure 8: The Gantt chart representation of the plan for the ALFA pilot.

simultaneously; therefore, the human worker should rotate the piece and, then, the operator and the robot unscrew bolts from two lateral sides of the piece. Figure 8 shows that the plan satisfy the production requirement of the high-level task. Indeed, the synchronization rule requires that the the low-level tasks for unscrewing bolts should be executed during the *BaseRemoval* task. Moreover, the first unscrew tasks must be performed *before* the operator rotates the piece, and the second unscrew tasks must be performed *after* the operator rotates the piece. It is also possible to observe that, robot's tasks are further decomposed in order to synthesize a more detailed representation of the activities the robot must perform to actually carry out the low-level tasks. For instance, the robot must set the arm on a specific target and then must activate the tool in order to perform an unscrew operation. Again, in Figure 8, a *during* temporal constraint holds between the *Unscrew* low-level task token and the *OnTarget* and *Operating* tokens.

Robust Task plan execution. Plans generated by the task planner can be temporally flexible, hence associated with an envelope of possible executions traces. This is a feature of APSI-TRF and EPSL that allow to be less brittle at execution being able to face temporal uncertainty in activities duration. Then, the actual execution of these plans is decided on the fly and, without an execution policy, a valid plan may fail due to wrong dispatching or environmental conditions (controllability problem (Vidal and Fargier 1999)). In order to address this issue, we leverage a recent research result (also integrated in KEEN) exploiting formal methods to generate a plan controller suitable for the execution of a flexible temporal plan (Orlandini et al. 2013). Namely, UPPAAL-TIGA, a model checker for Timed Game Automata (TGA), is exploited to synthesize robust execution controllers of flexible temporal plans. A TGA-based method for the generation of flexible plan controllers is integrated within the APSI-TRF Plan Execution module (Orlandini et al. 2013). In this case, the UPPAAL-TIGA engine is embedded within the planning and execution cycle generating plan controllers that guarantees a correct and robust execution. This is an important feature of the FourByThree task planning system as it enforces a safe plan execution further enforcing that all the production requirements and human preferences are properly respected.

Conclusions

The planning-based framework described above is to provide the control architecture with suitable deliberative features relying on the control model generated by the Knowledge Engineering according to the definition provided by the Production Engineer and the preferences of the Human Worker. An off-the-shelf planning and execution system based on APSI-TRF is then deployed to synthesize a suitable set of actions (i.e., in this work a timeline-based plan) that when executed controls the mechatronic device. It is worth reminding how the goal of the whole system is to create an environment to facilitate the task coordination between robot and human worker. In the near future we will be investigating how to gather information of execution of actions by human so as to have the system a comprehensive view of the plan execution.

Acknowledgment. The CNR authors are supported by the European Commission within the FourByThree project, GA No. 637095. Authors would also like to thank production engineers at ALFA and Inaki Maurtua and his colleagues at IK4-TEKNIKER for sharing with us the the production process of the case study reported in the paper.

References

Abdellatif, T.; Bensalem, S.; Combaz, J.; de Silva, L.; and Ingrand, F. 2012. Rigorous design of robot software: A formal component-based approach. *Robotics and Autonomous Systems* 60(12):1563–1578.

Barreiro, J.; Boyce, M.; Do, M.; Frank, J.; Iatauro, M.; Kichkaylo, T.; Morris, P.; Ong, J.; Remolina, E.; Smith, T.; and Smith, D. 2012. EUROPA: A Platform for AI Planning, Scheduling, Constraint Programming, and Optimization. In *ICKEPS 2012: the 4th Int. Competition on Knowledge Engineering for Planning and Scheduling*.

Borgo, S.; Cesta, A.; Orlandini, A.; and Umbrico, A. 2016. A planning-based architecture for a reconfigurable manufacturing system. In *The 26th International Conference on Automated Planning and Scheduling*.

Cashmore, M.; Fox, M.; Larkworthy, T.; Long, D.; and Magazzeni, D. 2014. AUV mission control via temporal planning. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, 6535–6541.

Cesta, A., and Fratini, S. 2008. The Timeline Representation Framework as a Planning and Scheduling Software Development Environment. In *PlanSIG-08. Proc. of the 27^{th} Workshop of the UK Planning and Scheduling Special Interest Group, Edinburgh, UK, December 11-12*.

Cesta, A.; Cortellessa, G.; Fratini, S.; and Oddi, A. 2009. Developing an End-to-End Planning Application from a Timeline Representation Framework. In *IAAI-09. Proc. of the 21^{st} Innovative Application of Artificial Intelligence Conference, Pasadena, CA, USA*.

Cesta, A.; Cortellessa, G.; Fratini, S.; and Oddi, A. 2011. MrSPOCK: Steps in Developing an End-to-End Space Application. *Computational Intelligence* 27(1).

Chien, S.; Tran, D.; Rabideau, G.; Schaffer, S.; Mandl, D.; and Frye, S. 2010. Timeline-Based Space Operations Scheduling with External Constraints. In *Proc. of the 20^{th} Int. Conf. on Automated Planning and Scheduling*.

Cialdea Mayer, M.; Orlandini, A.; and Umbrico, A. 2015. Planning and execution with flexible timelines: a formal account. *Acta Informatica* 1–32.

Cimatti, A.; Micheli, A.; and Roveri, M. 2013. Timelines with temporal uncertainty. In *AAAI*.

Fratini, S.; Cesta, A.; De Benidictis, R.; Orlandini, A.; and Rasconi, R. 2011. APSI-based deliberation in Goal Oriented Autonomous Controllers. In *ASTRA-11. 11th Symposium on Advanced Space Technologies in Robotics and Automation*.

Fratini, S.; Pecora, F.; and Cesta, A. 2008. Unifying Planning and Scheduling as Timelines in a Component-Based Perspective. *Archives of Control Sciences* 18(2):231–271.

Helms, E.; Schraft, R. D.; and Hägele, M. 2002. rob@work: Robot assistant in industrial environments. In *11th IEEE International Workshop on Robot and Human Interactive Communication*, 399–404. IEEE.

Laborie, P., and Ghallab, M. 1995. Ixtet: an integrated approach for plan generation and scheduling. In *Emerging Technologies and Factory Automation, 1995. ETFA '95, Proceedings., 1995 INRIA/IEEE Symposium on*, volume 1, 485–495 vol.1.

Lemaignan, S., and Alami, R. 2013. Explicit knowledge and the deliberative layer: Lessons learned. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, 5700–5707.

Marvel, J. A.; Falco, J.; and Marstio, I. 2015. Characterizing task-based human-robot collaboration safety in manufacturing. *IEEE Trans. Systems, Man, and Cybernetics: Systems* 45(2):260–275.

Muscettola, N. 1994. HSTS: Integrating Planning and Scheduling. In Zweben, M. and Fox, M.S., ed., *Intelligent Scheduling*. Morgan Kauffmann.

Orlandini, A.; Suriano, M.; Cesta, A.; and Finzi, A. 2013. Controller synthesis for safety critical planning. In *IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI 2013)*, 306–313. IEEE.

Orlandini, A.; Bernardi, G.; Cesta, A.; and Finzi, A. 2014. Planning meets verification and validation in a knowledge engineering environment. *Intelligenza Artificiale* 8(1):87–100.

Py, F.; Rajan, K.; and McGann, C. 2010. A Systematic Agent Framework for Situated Autonomous Systems. In *AAMAS-10. Proc. of the 9^{th} Int. Conf. on Autonomous Agents and Multiagent Systems*.

Sisbot, E. A., and Alami, R. 2012. A human-aware manipulation planner. *IEEE Trans. Robotics* 28(5):1045–1057.

Stanton, N. A. 2006. Hierarchical task analysis: Developments, applications, and extensions. *Applied Ergonomics* 37(1):55 – 79. Special Issue: Fundamental Reviews.

Umbrico, A.; Orlandini, A.; and Cialdea Mayer, M. 2015. Enriching a temporal planner with resources and a hierarchy-based heuristic. In *AI*IA 2015, Advances in Artificial Intelligence*. Springer. 410–423.

Vidal, T., and Fargier, H. 1999. Handling contingency in temporal constraint networks: from consistency to controllabilities. *Journal of Experimental and Theoretical Artificial Intelligence* 11:23–45.