Human Robot Collaboration in production environments

Christoph Strassmair[1] and Nick K. Taylor[2]

***Abstract*— Human Robot Collaboration has great potential for the manufacturing domain. A necessary precondition for this type of *Joint Action* is however the human's acceptance of their robotic co-worker. Recent research indicates that this acceptance is heavily influenced by the flexibility granted to the worker and the efficiency of the collaboration. Current systems aim for efficiency but neglect the spatial constraints of the collaboration. Furthermore they restrict the worker's flexibility. This paper presents an approach that extends prior art by considering spatial constraints and granting more flexibility to the worker. It thus facilitates a more efficient collaboration and a higher worker acceptance in manufacturing environments. The application of the approach to the workshop example is sketched.**

I. Introduction

Mobile assistive robots have great potential for manufacturing industry [1]. Compared to e.g. a domestic environment, the environments in focus are characterized by a higher level of structure, proceduralization and regulation. Locations of relevant tools and parts are organized according to ergonomic and economic aspects and are thus fixed. The tasks of the workers are time-constrained, embedded in the overall production process and may be very repetitive. In order to provide the timings, detailed task descriptions are created and consequently available. The actual task execution is regulated with respect to applying legal, health and safety terms, and the workers behavior is thus more predictable. These special conditions can ease problems which are hard to deal with in domestic environments.

Focusing on such environments, the potential of mobile robots grows as close physical collaboration with the human worker comes in reach. Of most importance for Human Robot Collaboration (HRC), and the resulting *Joint Action*, is the human's acceptance of the robot. With acceptance comes the willingness for collaboration, which is a necessary precondition for the success of the team. In a recent study on coordinating human-robot teams [2], an efficient collaboration that allows human flexibility regarding the task allocation achieved the highest satisfaction rating by the participants. Efficiency refers to the time required by the team to complete the task objective. Granting the humans flexibility requires the robot to adapt its behavior during the task execution according to the human co-worker. Consequently a dynamic task allocation and scheduling mechanism is required. Current approaches rely on a temporal formalism to realize such mechanisms. The temporal formalism enables the robot to reason about the temporal consequences of its actions. The second criterion for acceptance besides flexibility, the efficiency of the collaboration, can be achieved by formulating a scheduling policy for the robot actions, which is also enabled by the temporal formalism.

What current approaches neglect however, are the spatial constraints of such a close physical collaboration. Each adaptation of the task allocation or the schedule implies a different motion pattern of the team in the shared workspace. The result is changed spatial constraints for the robot, which effect the execution duration and thus the efficiency of the collaboration. Furthermore current approaches restrict the human flexibility by assuming the humans act only according to temporal "feasible" plans. Humans thus are supposed to have complete knowledge about the temporal consequences of their actions at each time of the collaboration. In order to overcome these shortcomings, this paper presents an approach that extends the prior art by integrating the spatial constraints in the task allocation and scheduling. Therefore the motion patterns in the shared workspace, implied by a concrete plan for collaborative task execution, are considered explicitly. As a consequence of the richer contextual description, a more robust and correct prediction of the temporal consequences of the robot actions is allowed. The improved predictions enable a more efficient behavior of the robot and thus a more efficient collaboration. Furthermore, the granted human flexibility is increased since the restrictions regarding the task allocation are eased.

The remainder of this paper is organized as follows. Section II describes related work. In section III basic concepts are reviewed, which are relevant to the presented approach. The workshop example and required modifications are presented in Section IV. The approach itself is described in section V before the paper is concluded by a discussion in Section VI.

II. Related Work

Whenever robots act in physical proximity to humans, the human behavior is influenced by the robot. The research area of Human Robot Proxemics (HRP) studies this phenomenon. Of special interest is the distance between human and robot in different situations. Especially for mobile robots meant to interact with the human the "proximic behaviour" of the robot "can make it socially acceptable and efficient" [3]. In [4] a planning framework and a planner are presented, generating such "social acceptable motions". HRP is however

[1]Christoph Strassmair is with the School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK. Furthermore he is with the Technische Hochschule Ingolstadt, Ingolstadt, Germany. christoph.strassmair@thi.de

[2]Nick K. Taylor is with the School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, UK. n.k.taylor@hw.ac.uk

not in focus of the presented approach. Due to the manufacturing environment and the instructed workforce, only minimal robot influence on the human behavior is assumed if the robot maintains a certain distance. This distance is ensured by the applied robot motion planning and can be adapted to newer findings.

Current systems realize HRC primarily in a turn-taking fashion, where each action of the robot is explicitly commanded by the human. The result is a stop-and-go collaboration, which cannot compete with a human team that fluently meshes the actions of the operators.

In [5] an "anticipatory action selection mechanism" was identified as key to this fluency. The presented mechanism allows the robot to control the timing of its actions. However, due to the fixed task assignment between the agents no significant difference in overall task efficiency could be measured. In [6], [7] an approach was presented that goes one step further and enables the robot to decide its next action as well as the timing. Inspired by human teaming behavior, the robot should be able to adapt to its human co-worker. In terms of adaptation, Human-Human Collaboration serves as a role model for Hoffman et al., Shah et al. and the approach presented here.

In order to ensure a valid and efficient action selection and scheduling, Shah et al. developed an Executive System (ES) called "Chaski" to control the robot. Chaski can operate in two different modes: "Equal partners" and "Leader & Assistant". In the Equal Partners mode both agents can freely choose their next action and its duration. In the Leader & Assistant mode only the leader has this freedom while the assistant tries to not constrain the leader. In both modes the agents are assumed to have complete knowledge about the temporal consequence of their choices, regarding the overall plan, and act only according to "feasible" plans. Feasible plans thereby refer to plans satisfying encoded temporal constraints as e.g. deadlines. In contrast, the approach presented here assumes that teams might execute non-feasible plans as well, resulting from the granted human flexibility. Consequently the execution of non-feasible plans is supported in order to increase the human flexibility.

The input of Chaski is a "multi-agent temporal plan". It specifies the number of agents, the actions to perform, the capabilities of the agents regarding doable actions and execution duration as well as a set of temporal and logical constraints relating the actions. The logical constraints are used to model precedence relations between actions. They can also be used to encode resource constraints and thus allow modeling spatial aspects. If the spatial issues that arise in shared workspace HRC, can be considered effectively, if modeled as logical constraints, remains unclear. Efforts towards applying Chaski in this direction have not been reported.

In [8] the multi-agent scheduling algorithm "Tercio" is presented. It enables dynamic task assignment and scheduling during runtime while respecting temporal as well as spatial constraints. Formulating a multi-agent task for Tercio includes setting an "allowable spatial proximity between agents". Similar to the approach presented here, each action of the task comes with a fixed location, where it has to be performed. By pairing these locations a set of pairings can be identified violating the allowable agent distance if performed simultaneously by the agents. Forbidding these pairings during the dynamic scheduling thus ensures the compliance with the allowable distance during action execution. While moving between the action locations this compliance is however not guaranteed. Furthermore the time required to move between action locations is not considered in the scheduling and thus endangers the temporal constraints.

III. Background

General problems of plan execution in the real world are uncertainties and disturbances. Temporal plans have been identified as a solution to robust and efficient execution. A temporal plan specifies "a partial order of actions with time information" [9]. It thus encodes a set of possible behaviours and enables an execution component to choose among them according to the actual execution conditions [10]. This type of dynamic scheduling is called *dispatchable execution*. Compared to a fixed time schedule, a "temporal plan allows the execution component to adjust to delays and fluctuations of action durations" [9]. This flexibility comes at the cost of constant adjustments of the plan during execution. The component choosing a concrete behaviour and performing the adjustments is called *Dispatcher*. In order minimize the time required in real-time for adjustments a second component, known as *Compiler*, transforms the initial temporal plan into an optimized form ahead of execution. The Compiler and Dispatcher together form an Executive or Executive System.

The time information of a temporal plan can be represented as temporal network. The simplest form of a temporal network is the Simple Temporal Network (STN) [11]. Each node in the STN represents an event or timepoint. The connecting edges represent constraints on the durations between the timepoints and are called links henceforth. Each link is associated with an interval describing the minimum and the maximum allowed time difference between the occurrences of the two connected timepoints. By interpreting the upper and lower bound of each interval as an individual edge, a STN can be transformed into a Distance Graph (DG) [11]. A STN is called *consistent* if the corresponding DG does not have any negative cycles. In the context of task scheduling, the consistency of a STN states that there is at least one schedule fulfilling all temporal constraints encoded in the underlying temporal plan. There is thus at least one way to execute the plan successfully. An exemple STN and its DG are shown in figure 1.

Also relevant to this work is the Simple Temporal Network with Uncertainty (STNU) [12]. The idea behind a STNU is the need in some domains to cope with external processes of uncontrollable, uncertain durations. In order to model this circumstance a STN is extended by further link and node types, called *contingent links* and *contingent timepoints*.

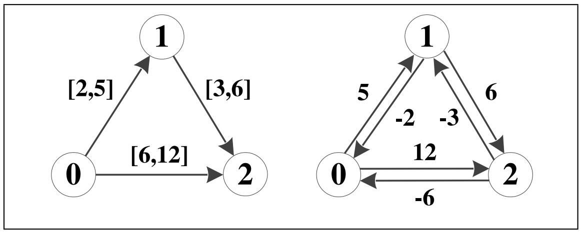

Fig. 1. Left: Exemple STN, Right: corresponding DG

Regular STN links are called *requirement links* in the STNU context. An example STNU is depicted in figure 2.

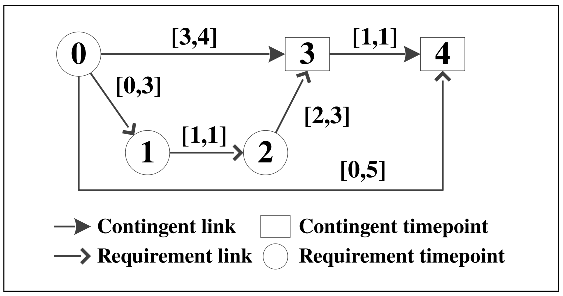

Fig. 2. Example STNU

Contingent links, like requirement links, are defined by an interval. Thus, though uncontrollable, the duration has to be specified by a lower and upper bound. Contingent links end in a contingent timepoint, which is "controlled by Nature[1] and subject to the limits imposed by the bounds on the contingent link" [13]. The remaining nodes are called *executable timepoints* and are under control of the executing entity. Regarding the execution of a STNU the property in question is controllability instead of consistency. By ignoring the semantic difference between the two link types, a STNU can be checked for consistency similar to a STN. Consistency however, is not sufficient to guarantee a successful execution as this would imply full control over all link values. Instead it has to be determined whether there exists a value selection strategy for only the requirement links, ensuring successful execution of the plan regardless of the Nature-chosen contingent link values. If such a strategy exists the STNU is considered controllable. Different levels of controllability have been identified. The relevant one for this work is called Dynamic Controllability (DC). If a STNU is DC, there exists a value selection strategy for only the requirement links ensuring a successful execution of the plan and, just as important, this strategy can be determined during execution.

IV. Workshop Example

The workshop organizers proposed the following example[2]: "A human and a robot have the goal to build a pile with 4 cubes and put a triangle at the top. One after the other, they should stack bricks in the expected order. Each agent has a number of cubes accessible in front of him and would participate to the task by placing its cubes on the pile. At the end, one of the agents should place a triangle at the top of the pile." The available actions for each agent are: take an object on the table, take an object from the pile, put an object on the pile, give an object to the other agent and support the pile. Object refers to either a cube or the triangle. Furthermore, each agent is able to infer the state of the world so it knows: where each object is, if an object is reachable for itself and if an object is reachable for the other one. Moreover, it's assumed that each agent is able to observe the activity of the other.

Since the presented approach focuses on tasks with temporal constraints and mobile systems, the example has to be modified slightly to apply the approach. First of all the cubes and the triangle are no longer available at the assembly table, but have to be fetched from different storages. This modification adds the path planning aspect of mobile systems. Furthermore each agent has access to all objects. The action 'give an object to the other agent' is thus superfluous. As a consequence the sub-actions of the piling step, 'take an object on the table' and 'put an object on the pile', can be merged to a single action 'fetch and put object on the pile'. Since each object has a different storage location, an individual action is required for each object. The used method of temporal planning requires an action sequence without loops. Hence agents are not allowed to remove objects from the pile; the action 'take an object from the pile' has to be dropped. Moreover, the approach was designed to cope explicitly with situations in that one agent is temporarily unavailable. Thus actions requiring both agents for execution are not in focus in the current state, although they are relevant and will be added in future work. As a consequence the action 'support the pile', which only makes sense if both agents work in parallel, cannot be considered at the moment. The new set of actions is:

1) fetch and put cube 1 on the pile
2) fetch and put cube 2 on the pile
3) fetch and put cube 3 on the pile
4) fetch and put cube 4 on the pile
5) fetch and put triangle on the pile

In order to apply the introduced approach to the example, additional information regarding the task is required. This information includes: a map of the workspace, action locations and durations as well as a total makespan deadline. For the missing information plausible data is added by the author. The resulting task description is pictured in the next section.

V. Approach

The presented approach is framed as an ES in the sense of [9], preparing the task offline and steering the execution online. The approach is thus split up into a Compiler and a Dispatcher.

[1]Nature refers here to the real world and subsumes external processes not under control of the network executor.

[2]http://fja2014.sciencesconf.org/resource/page/id/3

A. *Compiler*

The compiler takes a task as input and computes a set of HRC schedules, each describing a possible way that the human and robot could execute the task collaboratively. The task is specified by a task description, visualized for the workshop example in figure 3. The task graph as first element specifies precedence relations between the actions that have to be performed in order to complete the task. Due to the strict precedence relations set in the example, the resulting task graph is plain action sequence. The second element in the task description is a set of action descriptions that holds information about all actions involved in the task graph, as e.g. action locations and execution durations. The third element of the task description is the map that specifies the shared workspace for the HRC task. The sketched map shows a possible workshop setup. The numbers indicate the action locations of the five actions. 'H' and 'R' mark the initial human and robot position while 'T' represents the assembly table. The task description is completed by a set of temporal constraints, which further specify temporal relations between actions as e.g. deadlines. For the example a total makespan deadline, connecting the first and the last action, is pictured. For the environments in focus, the task description information should be available in the production planning system. Minor effort might be required for extracting and formatting the information. An overview of the compiling process is given in figure 4.

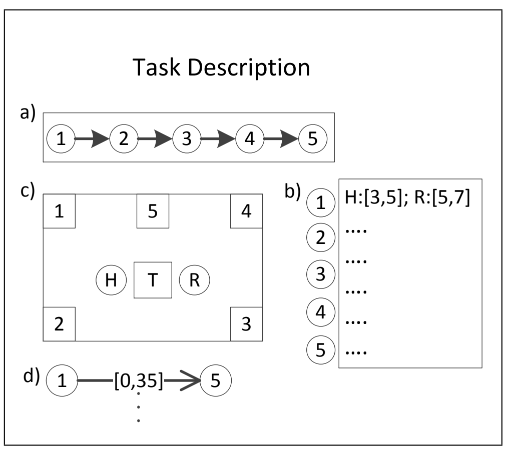

Fig. 3. Task Description: a) Task Graph, b) Action Descriptions, c) Workspace Map, d) Temporal Constraints

The first step performed in the compiler is called Plan Generation. It computes all plans describing a feasible way that the task could be executed collaboratively. A plan refers to a sequence of actions for each team member, satisfying the precedence constraints implied by the task graph. In order to obtain these plans all action allocations are computed in the first place. Subsequently for each allocation all valid sequences are calculated. The sum of all allocations and their valid sequences resembles the resulting set of plans.

As each action has to be performed on a specific location, each plan implies a motion pattern of the co-workers in the shared workspace. The durations of these motions vary from plan to plan as the co-workers influence each other: e.g. in a particular plan one worker might block the other's way and cause a detour. Obtaining these motion durations is done in the second step of the compiler called Motion Acquisition. The durations can be learned from observing human robot teams in the particular workspace or approximated by applying human and robot motion planners.

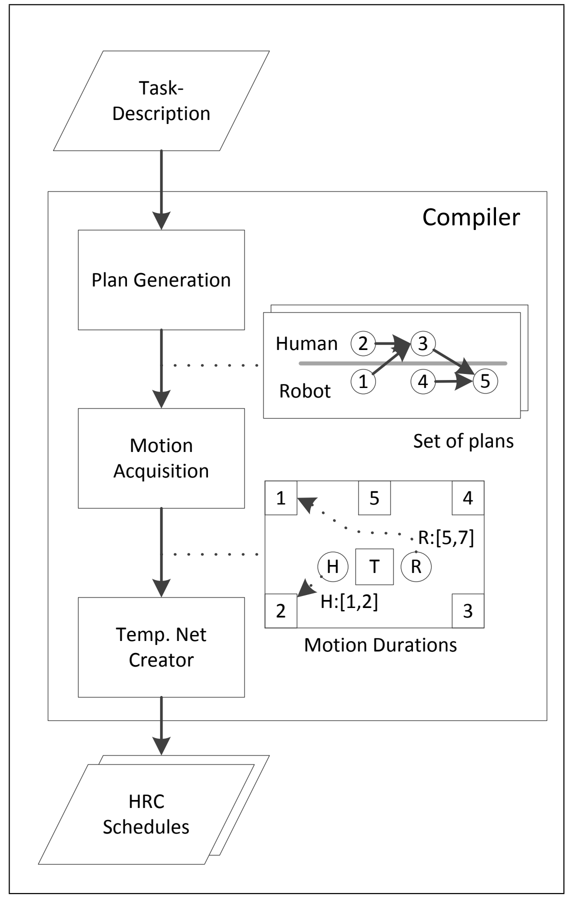

Fig. 4. Compiler: Input, computation, Output

The last step in the compiler is the construction of a HRC schedule for each plan. The schedules are encoded as STNUs and the module performing the encoding is called Temporal Net Creator. The reason for using the STNU formalism to represent the schedules is that the human is not under the control of the robot. Actions performed by the human thus are considered as external processes and consequently modeled as contingent links. The concrete duration of a human action is decided by the human, although it is assumed to be within the specified lower and upper bound. Robot actions are modeled as requirement links as their values are under control of the robot. The start and end node of a link represent the start and end timepoint of the action.

B. *Dispatcher*

The task of the Dispatcher is making real-time decisions regarding the robot's next action. The decisions are based on the offline computed set of HRC schedules. They are influenced by the behavior of the human co-worker during task execution, which is determined by the Human Action Detector. An action chosen by the Dispatcher is forwarded to the Robot Controller for execution. The completion of the action is acknowledged with a timestamp afterwards. The overall online process is shown in figure 5.

The Human Action Detector determines the human action choice and the execution timing. A lean implementation could rely on buttons or verbal commands, allowing the human to explicitly communicate the information. Implicit communication could be realized by applying a Hidden-Markov-Model, using the human position as observation and representing the human action as hidden state [14], [15]. The Dispatcher is cyclically updated by a message stating the currently performed and the predicted next action.

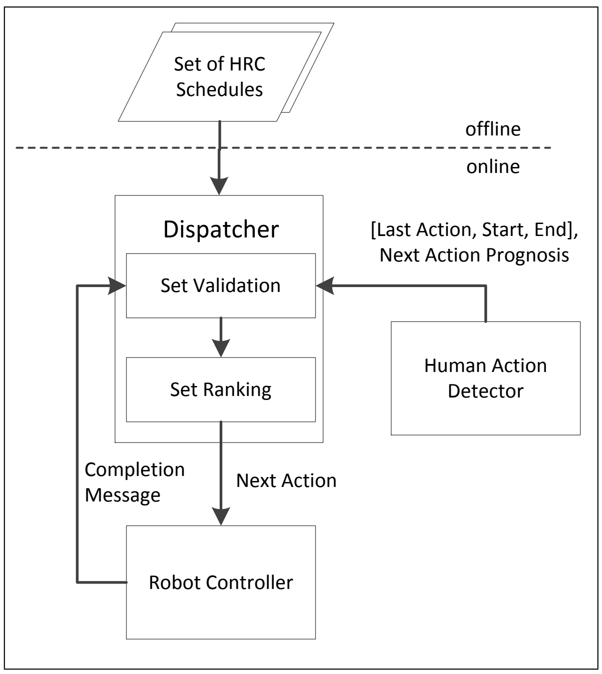

Fig. 5. Online Process

The set of HRC schedules computed offline provides the instructions for the collaborative task execution. Which concrete schedule the execution will follow depends on both agents. As the human has the freedom of choice regarding their actions[3], the determination of the concrete schedule in the Dispatcher is an iterative refinement process. The iterations are triggered by choice and accomplishment of human and robot actions. During each cycle the set of candidate HRC schedules is first updated, then validated and subsequently ranked.

The Set Validation module checks the validity of the current set of HRC schedules by matching the observed action allocation and sequence. Each schedule that does not match the observation is dropped since it does not represent the shared mental model of the team. Important to notice is that the observed timing of the actions is not considered in the validation step. While in the Chaski system of Shah et al. plans violating temporal constraints are not added to the solution set, they represent possible solutions in this approach. The reason is the flexibility granted to the human on the one hand and the goal to complete the task on the other hand. As a consequence of the human's choice the task completion might violate temporal constraints. This particular choice is encoded in a particular plan, which is non-feasible due to the violated constraints. Dropping the plan however would mean dropping the instructions for this particular choice. The robot thus would not be able to complete the task, which is even worse than not complying with the deadline.

The timing of the observed actions is however not neglected. Instead of using it for the validation, it is used for a ranking of the schedules in the Set Ranking module. After updating the available schedules by the observed timing, they are verified according to the specified temporal constraints. The verification is thereby focused on the two central properties introduced in section III: Dynamic Controllability (DC) and Consistency. The schedules are encoded as STNUs. A property of great importance for executing a STNU is DC [13], [16]. In the context of this work the DC property of an STNU states that there exists an execution strategy for the robot, which ensures task completion while satisfying the temporal constraints. Consequently, as long as the human performs their activities within the specified bounds, successful task completion can be ensured solely by the robot for a DC schedule. A schedule however can also be interpreted as STN. The semantic meaning of this interpretation is that human and robot as a team are responsible for the execution. Together they have full control, making the distinction between contingent and requirement links to represent controllable and uncontrollable durations irrelevant. Interpreting the schedule as STN makes the Consistency property the relevant one. A STN is called consistent if it has at least one solution. In the context of this work Consistency states that there is at least one strategy for the human-robot-team, which ensures task completion while satisfying the constraints. Ensuring successful task completion of a consistent schedule thus cannot be done by the robot alone but relies also on the human behavior.

The ranking of the schedules is based on three categories and the introduced properties. The highest ranked schedules are the dynamic controllable ones as the robot can ensure a successful execution. The next category holds the consistent schedules, depending additionally on the human choices, regarding a successful execution. The lowest ranked schedules are the inconsistent ones that can only be executed successfully if activities are performed faster than assumed by the lower bounds. Within each category additional metrics, as e.g. total duration (makespan) of the schedule or human and/or robot idle time, can be applied to refine the ranking. The best ranked schedule of the highest non-empty category determines the next action for the robot.

[3]Regarding the human's next action decision a non-preemptive behavior is assumed: a next action choice is not reverted and the action is performed until completion.

VI. DISCUSSION

We have presented an approach that extends the prior art regarding task allocation and scheduling for close physical collaboration of human and mobile robot in manufacturing environments. The approach integrates spatial constraints inherent in this type of collaboration. This allows the robot a more robust and correct prediction of the temporal consequences of its actions, which enables a more efficient robot behavior and lastly a better acceptance of the collaboration by the human. Furthermore, the flexibility granted to the human regarding the task allocation is increased. This is achieved by also considering allocations, and resulting schedules, that potentially violate temporal constraints. Since task completion is regarded as most important, the violation of temporal constraints is tolerated if task completion is endangered. The resulting increase of allowed human flexibility enables a better acceptance of the collaboration by the human as well. The central hypotheses underlying the approach are twofold. The first one is the positive influence of considering the spatial constraints on the overall efficiency. The second one is a higher human satisfaction in the collaboration due to the increased allowed human flexibility. Both will be evaluated in future work.

Applying the approach to the workshop example requires several modifications due to different underlying assumptions. The presented approach was developed for deterministic, durative actions in a dynamic environment. The example description however rather implies non-deterministic, instantaneous actions in a static environment. As consequence of these differences incorrect behavior of an agent, failing actions or collapsing piles cannot be handled by the presented approach. A negotiation phase is not considered in the approach since granting flexibility to the human is an explicit goal. Thus the robot adapts its behavior in case of conflicting intensions, while considering the temporal constraints. Similarly situations of inactiveness are dealt with: flexibility is granted to the human and the robot acts with respect to the temporal constraints. In case the human is temporarily unavailable, the robot is meant to continue the task on its own.

REFERENCES

[1] S. Angerer, C. Strassmair, M. Staehr, M. Roettenbacher, and N. M. Robertson, "Give me a hand - The potential of mobile assistive robots in automotive logistics and assembly applications," in *Technologies for Practical Robot Applications (TePRA), 2012 IEEE International Conference on*, 2012, pp. 111–116.

[2] M. C. Gombolay, R. J. Wilcox, A. Diaz, F.. Yu, and J. A. Shah, "Towards Successful Coordination of Human and Robotic Work using Automated Scheduling Tools: An Initial Pilot Study," in *Proceedings of Robotics: Science and Systems, Human-Robot Collaboration Workshop*, 2013.

[3] M. Asghari Oskoei, Michael Walters, and Kerstin Dautenhahn, "An Autonomous Proxemic System for a Mobile Companion Robot," in *Proceedings of the AISB 2010 Symposium on New Frontiers for Human Robot Interaction*. AISB, 2010, pp. 9–15.

[4] E. A. Sisbot and R. Alami, "A Human-Aware Manipulation Planner: Robotics, IEEE Transactions on," *Robotics, IEEE Transactions on*, vol. 28, no. 5, pp. 1045–1057, 2012.

[5] G. Hoffman and C. Breazeal, "Cost-Based Anticipatory Action Selection for Human-Robot Fluency," *IEEE Transactions on Robotics*, vol. 23,, no. 5, pp. 952–961, 2007.

[6] J. A. Shah, J. Wiken, B. C. Williams, and C. Breazeal, "Improved human-robot team performance using chaski, a human-inspired plan execution system," in *Proceedings of the 6th International Conference on Human Robot Interaction, HRI 2011, Lausanne, Switzerland, March 6-9, 2011*. ACM, 2011, pp. 29–36.

[7] J. A. Shah, B. C. Williams, and C. Breazeal, "Dynamic Execution of Temporal Plans for Temporally Fluid Human-Robot Teaming," in *It's All in the Timing, Papers from the 2010 AAAI Spring Symposium, Technical Report SS-10-06, Stanford, California, USA, March 22-24, 2010*. AAAI, 2010.

[8] M. Gombolay, R. Wilcox, and J. Shah, "Fast Scheduling of Multi-Robot Teams with Temporospatial Constraints," in *Proceedings of Robotics: Science and Systems*, 2013.

[9] N. Muscettola, P. H. Morris, and I. Tsamardinos, "Reformulating Temporal Plans for Efficient Execution," in *Principles of Knowledge Representation and Reasoning*, 1998, pp. 444–452.

[10] I. Tsamardinos, N. Muscettola, and P. Morris, "Fast transformation of temporal plans for efficient execution," in *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*. American Association for Artificial Intelligence, 1998, pp. 254–261.

[11] R. Dechter, I. Meiri, and J. Pearl, "Temporal constraint networks," *Artif. Intell*, vol. 49, no. 1-3, pp. 61–95, 1991.

[12] T. Vidal, "Handling contingency in temporal constraint networks: from consistency to controllabilities," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 11, no. 1, pp. 23–45, 1999.

[13] P. Morris, N. Muscettola, and T. Vidal, "Dynamic control of plans with temporal uncertainty," in *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 1*. Morgan Kaufmann Publishers Inc, 2001, pp. 494–499.

[14] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning Motion Patterns of People for Compliant Robot Motion," *International Journal of Robotics Research*, vol. 24, no. 1, 2005.

[15] D. Kulić and Y. Nakamura, "Incremental learning of human behaviors using hierarchical hidden Markov models," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 4649–4655.

[16] L. Hunsberger, "Fixing the Semantics for Dynamic Controllability and Providing a More Practical Characterization of Dynamic Execution Strategies," in *16th International Symposium on : Temporal Representation and Reasoning, 2009. TIME 2009.*, 2009, pp. 155–162.