Assembly and Task Planning Using Petri Nets: A Survey*

Jan Rosell

Institute of Industrial and Control Engineering,

Technical University of Catalonia, Barcelona, SPAIN

Abstract:

The automatic synthesis of a plan to perform a robotized assembly task from a high level description of the product to be assembled is a difficult issue which involves assembly planning and task planning. This paper summarises the problems that an autonomous robotic assembly system must tackle both regarding assembly planning and task planning, and reviews the approaches that use Petri Nets as a formalism to develop the corresponding planners. The need of integration of both assembly and task planning is made evident, and approaches covering that are described.

Keywords: Assembly planning, task planning, Petri nets

1 ASSEMBLY AND TASK PLANNING

The field of assembly and task planning addresses the issue of how to synthesise a plan for a robot to execute an assembly task from a high level description of the product to be assembled. Following over two decades of research, the IEEE Technical Committee on Assembly and Task Planning presented a taxonomy on the field [1].

* This work was partially supported by the CICYT projects DPI2002-03540 and DPI2001-2202

Assembly planning is generally associated with sequence planning, i.e. the determination of a (feasible and optimal) sequence of operations (e.g. part matting operations) to be done in order to assemble a product. This sequence planning problem, however, must be considered in conjunction with the design process (for concurrent engineering), the workcell planning (to take into account the availability of resources), and the assembly representation issues (e.g. how to handle tolerances and uncertainties).

Task planning deals with the translation of an assembly plan into the robot operations that may allow the successful execution of the task. Task planning involves the planning of sensory operations (e.g. planning of visual sensing operations), gross-motion planning (the planning of the robot motions in order to avoid collisions), and fine-motion planning (the planning of the robot motions in order to execute part-matting operations). Fine-motion planning requires taking into account uncertainties and tolerances, and involves the planning of compliant motions (using force and torque sensory feedback).

Finally, in order to execute the task, code generation must also be carried out by the task planner. This is a difficult issue, since advanced robotic tasks involve concurrency, synchronization and real-time requirements. Therefore, the use of formal methods in the whole development cycle is required to allow the verification of functional and temporal requirements, yielding to reliable control software. Early validation of the design of the control software, as well as the use of an automatic tool in the code phase, are crucial to avoid the catastrophic effects of a failure.

Petri nets are a modelling formalism that has been used in both assembly planning and task planning. This paper makes an introduction to Petri nets (Section 2) and presents a review of the approaches that have used it for assembly planning (Section 3) and task planning (Section 4). The need of integrating both is made evident in Section 5. Finally, Section 6 summarizes the work by reviewing the main objectives and requirements posed by an autonomous robotic assembly system, and how Petri nets contribute to model and fulfil them.

2 PETRI NETS FORMALISM

Petri nets are a formalism that allows the modelling of systems involving concurrency, resource sharing, synchronization and conflict, and allows the validation of the correctness of the system by analyzing the qualitative properties of the net modelling the system.

A Petri net model of a dynamic system is composed of a net structure that represents the static part of the system, and a marking that represents a distributed overall state on the structure. The net structure is a weighted-bipartite directed graph specified as a four tuple [2,3]:

$$N = < \boldsymbol{P}, \boldsymbol{T}, F, W >$$

where:

- $\boldsymbol{P}$ is a finite non-empty set of places, graphically represented as circles.
- $\boldsymbol{T}$ is a finite non-empty set of transitions ($\boldsymbol{P} \cap \boldsymbol{T} = \emptyset$), graphically represented as bars.
- F is a set of directed arcs $F \subseteq (\boldsymbol{P} \times \boldsymbol{T}) \cup (\boldsymbol{T} \times \boldsymbol{P})$
- $W : F \rightarrow \aleph^{+}$ is a function assigning a weight to each arc ($\omega(t, p)$ is the weight of the arc from transition *t* to place *p*, and $\omega(p, t)$ is the weight of the arc from *p* to *t*). The weight is graphically represented as a label on the arc, although it is usually omitted if it is equal to one.

As an example Fig. 1 shows a Petri net illustrating the modelling of concurrency and synchronization (activities represented by p_4 and p_5 are executed in parallel and are synchronized at t_3), and resource sharing and conflict (p_2 is a shared resource used either by p_4-p_5 or by p_6, the choice being an open conflict).

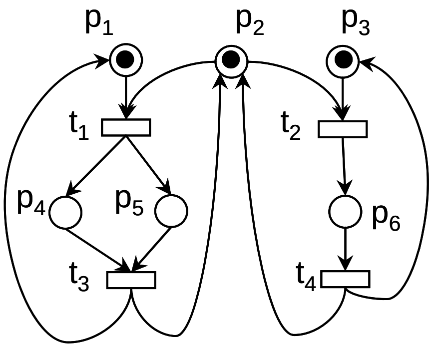

Fig. 1: A Petri net composed of six places and four transitions.

The marking of a net N is an application of $\boldsymbol{P}$ on $\aleph^+$ (i.e. an assignment of tokens to each place, graphically represented as dots inside the place). The marking changes according to a marking evolution rule (known as token game) that describes the system state changes (Fig. 2):

1. A transition t is enabled if every input place p_i of t is marked with at least as many tokens as specified by the weight $\omega(p_i, t)$.
2. An enabled transition t may fire. The firing of t removes $\omega(p_i, t)$ tokens from each input place p_i of t, and adds $\omega(t, p_o)$ tokens to each output place p_o of t.
3.

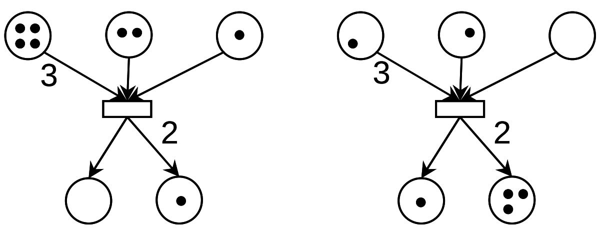

Fig. 2: Enabled transition (left) and the marking obtained after its firing (right).

Formal methods have been developed that allows the analysis of the following qualitative properties of Petri nets:

a) boundedness, which characterizes the finiteness of the state space,
b) liveness, which characterizes the possibility of any event described by a transition to occur from any reachable marking, and
c) reversibility, which characterizes the recoverability of the initial marking from any reachable one.

One of the most used methods is based on the reachability tree, which is a tree whose nodes are the markings that are reachable from the initial marking, the arcs being labelled by the transition to be fired to change from one marking to the next.

The strength of the Petri nets formalism is the combination of an expressive graphical specification together with a well formalized mathematical model.

2.1 High Level Petri Nets (HPN)

Ordinary Petri nets suffer, however, from the lack of high-level constructs, which affects the readability of the final design. To cope with this problem, high level Petri nets (HPN), like coloured nets (CPN) [4] or predicate transition nets (PrTN) [5] extended ordinary Petri nets by:

a) Attaching data values to net tokens (which can be integers, reals, or complex data types like products, records or lists, and are called colours).

b) Using variables in arc expressions.

c) Attaching boolean expressions (guards) to transitions.

These extensions allow the use of much fewer places and transitions than needed in ordinary Petri nets. Further, HPN can also be extended by a time concept that permits to describe the duration of actions.

2.2 Hierarchical High Level Petri Nets (HHPN)

Hierarchical high level Petri nets [6,7] allow the system modeller to describe a set of submodels which all contribute to a much larger model by interacting with each other in a well-defined way. Either a place or a transition can be a substitution node of the net by associating a submodel to it. Substitution nodes are not themselves part of the global model since they only describe how the associated submodel is related to the global model.

During analysis of complex systems it is convenient to temporarily ignore some parts of a model or replace them with simple components. This can be done if substitution nodes become ordinary nodes, the change being possible since substitution nodes with their surrounding arcs are behaviourally equivalent to the related submodels.

2.3 Computer Based Tools

The need of computer based tools for the definition and analysis of Petri net models is evident. A very comprehensive list of Petri Net tools can be found in *http://www.daimi.au.dk/PetriNets/tools/*. An evaluation and comparison of some of these available tools for HPN can be found in [8,9]. One of the best is Design/CPN [10] due to the high capacity analysis provided. Design/CPN has a graphical editor (that allows the user to construct, modify and syntax check hierarchical coloured Petri nets, with or without time), a simulator (that allows their simulation, either interactively or in a fully automatic way), and an analysis tool (that allows the user to construct and analyze the reachability tree).

3 ASSEMBLY PLANNING USING PETRI NETS

This section deals with the use of Petri nets in assembly planning. Petri nets eases the representation of the subtasks into which an assembly can be decomposed taking into account the associated pre-conditions and post-conditions (precedences, material and facility requirements, stability of subassemblies,...) which determine the feasible sequences. The qualitative analysis of the Petri net models allows the study of correctness of the assembly system. The structural properties of the Petri net determine the search strategies.

A related problem that has recently received attention is that of disassembly planning, because over the last few years there has been increasing pressure to produce and dispose of products in an environmentally responsible way. Despite the similarities of assembly and disassembly, it is necessary to deal with these differently due to several unique aspects that appear in disassembly. For example, the uncertainty in the structure and component conditions for disassembly results in an uncertain termination goal [11]. Petri nets have proved to be a very useful tool in this field [12].

This section also includes the use of Petri nets in the study of assembly systems from a wider perspective, i.e. the planning of the flexible assembly system (FAS).

3.1 Assembly planning

Petri nets were first used for assembly planning by Zhang [13], which used transitions and places of a Petri net to represent the assembly operations and the corresponding preconditions and results, respectively. The assembly plan was generated by consecutively firing the enabled transitions, using heuristic rules to solve potential conflicts between simultaneously enabled transitions.

The need of a formal definition of all possible assembly plans of a given product lead Homem de Mello and Sanderson to propose the AND/OR graphs [14]. The nodes of this graph are the subsets of the assembly that correspond to connected-stable subassemblies. The arcs represent the physically feasible decomposition of subassemblies into smaller subassemblies. OR-relations define the different methods in which a given subassembly can be decomposed; AND-relations describe the subassemblies obtained. AND/OR graphs can show explicitly the possibility of parallel execution of assembly operations and the time independence of operations that can be executed in parallel. As a consequence the AND/OR graphs are a useful tool for the selection of the best assembly plan, the recovery of execution errors and the opportunistic schedule of tasks. Later, Cao and Sanderson [15] described an extension of AND/OR graphs from a tree to a net structure, and presented a mapping from the AND/OR net representation of an assembly to a Petri net, in order to ease the analysis of the qualitative properties of the system. The Petri net obtained was demonstrated to be bounded, live and reversible. The searching of assembly sequences was done either from the AND/OR net by constructing a state graph, or from the Petri net by analyzing the corresponding reachability tree, using a searching algorithm similar to Dijkstra. Figure 3 illustrates the Petri net model of a very simple assembly product.

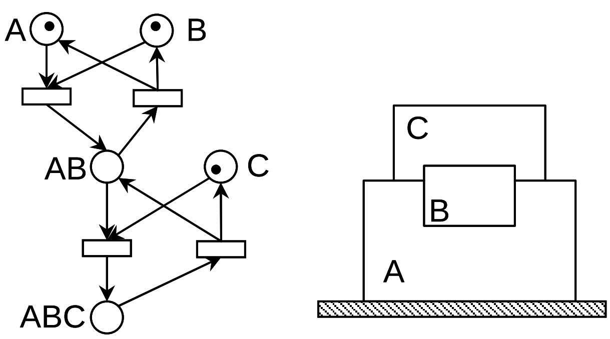

Fig. 3: The Petri net model capturing the relations between the parts of an assembly product.

Petri nets obtained from AND/OR nets were also demonstrated to satisfy that the optimal task sequence can be obtained by linear programming techniques instead of most costly searching algorithm, due to the algebraic and structural properties of such nets [16].

In order to model uncertain situations, fuzzy information has been incorporated into ordinary Petri nets [17,18]. For example, a fuzzy marking variable allows the uncertain localization of the objects in the system to be described.

CPNs have also been used for assembly planning. In [19] a catalogue of CPN modules is presented to model basic cell operations (pick, place, part-mating, store, retrieve, recognition, convey, feeding and unloading). Mating rules (operation chains, OR-schemes, resource allocation/releasing, test for conditions) and reduction rules (merging transitions, merging places, reducing OR-schemes) are also proposed for the construction of the system model from the CPN modules. This CPN model is the basis of an assembly planning algorithm not reported in the paper.

CPN models are used in [20] within the scope of flexible assembly systems and assembly job shops. The authors argue that using CPN models, assembly plans are implicitly generated with a degree of parallelism higher than using AND/OR graphs. A main CPN decomposition module is proposed, which describes the disassembly process of a subassembly into two smaller subassemblies, and colours are defined to represent the relation between subassemblies. Off-line planning is performed by a deep exploration of the different solutions of the conflicts among the colours of the tokens that simultaneously enable a transition of the CPN.

3.2 Disassembly planning

Disassembly process planning is critical for minimizing the resources invested in disassembly and maximizing the level of automation of the disassembly process and the quality of the parts recovered. Zussman and Zhou [11,21] proposed a Disassembly Petri Net (DPN) for the modelling and adaptive planning of disassembly processes. Their approach is an adaptive process-planning scheme that is able to introduce process changes in order to comply with specific product conditions. DPN places model subassemblies or parts, and have an associated function characterizing its end of life value. DPN transitions model disassembly operations and have an associated function characterizing the disassembly cost. Transitions also have probability values (updated during process execution) which represent the success rate due to uncertainty caused by different product conditions and performance of the external resources. Online adaptive planning is performed using decision values (based on the above two functions) and the probability values.

Moore et al. [22] proposed an algorithm that automatically generates a disassembly Petri net from a geometrically-based disassembly precedence matrix, computed from a CAD representation of the product. The proposed algorithm can be used for products containing AND, OR, complex AND/OR and XOR relationships and guarantees that, with a small number of restrictions, the obtained disassembly Petri Net is live, bounded and reversible. Near-optimal disassembly sequences are obtained from the corresponding reachability tree using a heuristic search approach.

The resources required in the disassembly facility have been considered in Tang et al. [12].

3.3 FAS planning

When the availability of assembly resources is different from the one planned, off-line computed assembly plans must be on-line modified. Therefore, system flexibility and uncertainty in assembly operations must be taken into account in assembly planning.

FASs are based on programmable, multi-functional facilities that allow different part routings and multi-product manufacturing. This poses synchronization problems which occur dynamically and require on-line decision making. Several works can be found that use CPN as a modelling methodology for flexible manufacturing systems, e.g. [23] presented a CPN model of a cell controller embedded in a CIM architecture to carry out scheduling tasks (loading, routing, dispatching). A hierarchical structure is proposed in [24] for the CPN modelling of a flexible assembly system, composed of three parallel and cooperative hierarchies: a material hierarchy (parts, subassemblies and products), a task hierarchy (assembly operations, parts transfers,…) and a resource hierarchy (equipment, tools,…). Along a similar but more formal line Borusan [25] proposes a systematic methodology for the construction of hierarchical CPN models for flexible manufacturing systems. It is argued [26] that CPN models are a useful tool in the life cycle of a FMS (modelling, analysis, simulation and control).

In some flexible assembly systems, uncertainties may result in the unavailability of resources that may lead to deadlocks which have a negative impact on the scheduled activities and performance. This problem is tackled in [27] by introducing Controlled Assembly Petri Nets (CAPNs) and a deadlock avoidance control logic that incorporates or removes production processes while maintaining the liveness property of the system. To take into account uncertainty, Shiu et al. [28] introduced a modelling methodology that integrates the modelling capability of CPN with the modelling capability of fuzzy Petri nets.

4 TASK PLANNING USING PETRI NETS

This section deals with the use of Petri nets in task planning. Task planning is devoted to the planning of the robot motions to execute the subtasks of a task sequence computed by an assembly planner. Subtask's execution usually involves contact motions through a sequence of contact states [29]. Due to uncertainties and manufacturing tolerances, the reachability of the goal is not guaranteed. Petri nets can capture the contact states of the task and the transitions between them, in order to plan a nominal solution sequence and recovery strategies.

Task planning also involves control implementation issues in order to execute the robotic assembly task. This implies the specification, validation and code-generation, which may be a hard and error-prone process. Petri nets are a formal methodology that is also useful in this field, which speeds up the process and reduces errors.

4.1 Task planning

Assembly operations can be decomposed into skill primitives to be executed by the robot. This is done in [30] where skills are defined as elementary sensor-based robot movements to reach predefined goal states (e.g. move-to-touch, rotate-to-level, rotate-to-insert). Assembly operations are categorized into different robot tasks depending on the type of operation, the pair of subassemblies involved, their symbolic spatial relations, the local depart space, and the tool used for the task. Each robot task must be performed as a sequence of skill primitives.

Other approaches directly map an assembly operation to a sequence of contact states between the manipulated object and the environment. Generally, assembly tasks involve the manipulation of polyhedral rigid objects. Under this assumption, task planning approaches usually rely on a graph representation of the contact space (the space composed of the configurations of the manipulated object that produce contact with the objects in the environment) that describes the possible contact states and their neighbouring relations.

McCarragher [31] presented the first Petri net based approach to the task-level control of robotic assembly. Assembly is modelled with a Petri net, where places are either state places or control places. State places represent basic contacts of the assembly (the simultaneous marking of state places representing multi-contact situation), and transitions represent the gaining or losing of a basic contact. Control places represent conditions of external inputs that enable transitions to occur, and are connected to each transition by an inhibitor arc. A trajectory planner defines a performance measure and finds a path in the Petri net (i.e. a sequence of transition firings) that maximizes it. Figure 4 shows a part of a Petri net that illustrates the possible transitions between two basic contacts of a peg-in-hole assembly task.

The automatic generation of the contact space graph was first proposed by Xiao and Ji [32], and later on by Rosell [33] using CPNs.

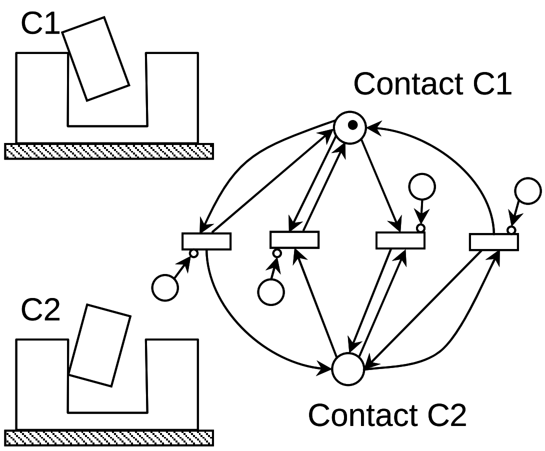

Fig. 4: Part of a Petri net model that captures the possible transitions between two basic contacts of a peg-in-hole assembly task

4.2 Control

The execution of robotic assembly tasks requires real-time closed-loop control programs. To cope with this, Simon [34] decomposed complex robot actions into subactions called robot tasks (RT), i.e. closed-loop control laws encapsulated in a reactive logical behaviour. RT are defined in terms of communicating real-time computing tasks called module tasks (MT) which implement an elementary part of the control law. The temporal attributes of MT are the duration, the activation period, the input and output ports and the priority. An appropriate synchronization can enforce MT's execution ordering compatible with data and time dependencies between algorithmic modules. Based on a Petri net representation, a method is proposed that allows an analysis of the behaviour of the RT's that guarantee that they are deadlock free. Figure 5 shows a simple example of a Petri net that represents the management of a shared memory that may be read or written by *k* processes (parallel program execution threads).

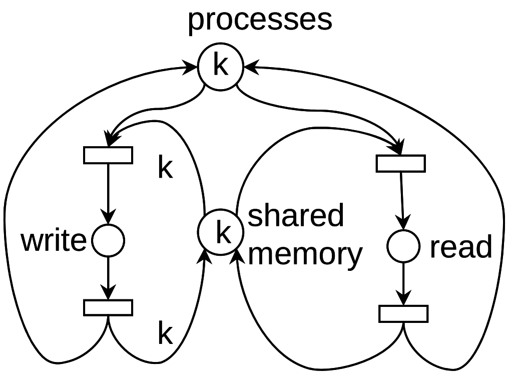

Figure 5: Petri net that represents the management of a shared memory that may be read or written by k processes (parallel program execution threads).

Synchronization in multi-robot systems is tackled in Suh et al. [35], based on the Petri net Graphical Robot Language (PGRL) [36]. In PGRL, each place represents a task command. By using the fork and the join of a Petri net, PGRL is able to represent a multi-robot job requiring parallelism, preconditions and synchronization between multi-manipulator motions. PGRL includes a program analyzer based on the reachability tree to guarantee the qualitative properties of the net.

Montano et al. [37] used time Petri nets (TPN) to develop a robotic control system. TPN incorporate time by associating a firing delay of each transition specified as an interval, which enable the modelling of time-outs, periodical activities, synchronization and concurrency. Their approach emphasises the implementation stages: real-time analysis, planning and code generation.

Code generation is also tackled in [38], which describes a general method for automatic code generation from coloured Petri nets, supported by the Design/CPN tool. The code, in the form of ML source [39], is extracted from the Design/CPN simulator. The ML source can be compiled by different ML compilers depending on the target hardware platform.

Automatic code generation for Programmable Logic Controllers is presented in [40,41], using an extension to classical CPN, called Ordered CPN (OCPN), that map the CPN model into sensor/actuator interfaces. Exchange of physical signals between CPN-model and the production environment is implemented as *sensor* transition guards. Conflicts are solved with the aid of a real-time decision system (scheduler), whose results are used through *scheduling* transition guards. The OPCN model is translated to code according to IEC 1131 standard.

In a higher control level, Moody and Antsaklis [42] dealt with the design of DES supervisors that ensure that the behaviour of the plant does not violate a set of constraints under a variety of operating conditions. The plant and the controller are both represented as Petri nets. The design of the controller is based on the fact that the specifications representing desired plant behaviours can be enforced by making them invariants of the feedback controlled system.

5 INTEGRATING ASSEMBLY AND TASK PLANNING

There is the need for integration of assembly planning and task planning. On the one hand, system flexibility and uncertainty in assembly operations give rise to the need for dynamic assembly planning, i.e. the need to change the assembly plan as a consequence of the task plan results. On the other hand task plans must be able to be updated according to assembly plan changes. Guéré and Alami [43] discuss this problem and argue that a generic planner must capture and learn the structure of a task domain since the environment may change on the actions that can be performed, i.e. the planner must be based on a graph of the task space that integrates abstract reasoning and geometrical constraints.

Song et al. [44] also discuss this issue. They argue that the efficiency, reliability and safety of robotic manufacturing systems can only be obtained by integrating task-scheduling, action-planning and control. Their proposal is based on the use of a max-plus algebra model and an event-based planning methodology. An action based variable is used that carries the sensory information needed for the planner to adjust the original plan, i.e. planning is made a real-time process.

An integrated planner should be described using the same modelling technique in all of its phases. Petri nets, as shown in the previous sections, have been used for both assembly planning and task planning. The first Petri net based approach to integrate both assembly planning and task planning was that of Cao and Sanderson [45]. They refined the Petri nets obtained from the AND/OR graphs by including the constraints imposed by the use of sensors and plans required for the execution of gross, fine, and grasp motions. This decomposition preserved the net properties and thus guaranteed a deadlock-free and fault-tolerant system.

Later, Thomas et al. [46] presented a unified hierarchical Petri net framework. In the assembly planning level, Assembly Petri Net (APN) are defined in a similar way as [15], having the properties of liveness and boundedness. Control plans are obtained from APN and the production system model following a refinement procedure (each task in APN is refined into a number of control tasks which are represented as a control plan subnet), and a merging procedure (control subnets are merged into a single control plan).

6 SUMMARY

The objectives and requirements posed by an autonomous robotic assembly system are summarized below. Their fulfilment when using Petri nets as a modelling methodology is discussed, and the research efforts still required are pointed out.

1. *Use of a unified modelling tool.* There are several problems at different levels of abstraction, from assembly planning to task planning (e.g. from the generation of the assembly sequence to the real-time control of the robot motions). These problems are coupled and, therefore, the use of the same modelling technique to model the system at its different levels of abstraction is a must for the comprehension of the whole system and the resolution of conflicts and interactions between levels.

   Petri nets have been successfully used in both assembly planning and task planning. Moreover, some approaches already proposed a hierarchical and modular methodology to integrate both. Efforts should be directed to systematize the modelling procedures, and to automate the planning process from a high level description of the task.

2. *Consideration of both assembly and disassembly.* An autonomous robotic assembly system must be able to cope with both assembly and disassembly tasks, and also consider complex relations between subassemblies. Assembly is usually based on the assembly from disassembly perspective. Therefore, a unified framework where both assembly and disassembly can be described seems a useful approach.

   Petri nets have been successfully used in both assembly and disassembly planning, although no unified approach has yet been proposed. Several aspects introduced by research authors in the field of disassembly (e.g. operations costs or actions uncertainty) seems to be also useful in assembly. This is made more evident when assembly planning is not considered as an isolated level but coupled with task planning issues. Therefore, a Petri net based framework for both assembly and disassembly planning may be an interesting contribution to the field.

3. *A dynamic and adaptive behaviour.* Uncertainty in the results of robot actions must be considered and properly handled, giving rise to the possibility of a dynamic planning and

execution of the task. Adaptive planning methods must be based on on-line update of performance data (from which several measures of quality can be defined).

Although some Petri net based approaches consider the integration of both assembly planning and task planning, some efforts are still required in order to be able to make assembly planning a real-time process.

4. *System flexibility.* Flexible assembly systems must be considered and the case of multirobot environments needs to be taken into account. An autonomous robotic assembly system must correctly handle the power of system flexibility, since the availability of resources influences the qualitative properties and the performance of the task.

   Petri nets have been successfully used in the analysis of many flexible manufacturing systems. They have also been used to analyze the impact of the unavailability of resources in the scheduled activities, like the actions in an assembly plan. This must be further investigated when task planning issues are also considered.

5. *Need of a systematic modelling procedure.* The modelling procedure must ensure a correct system (i.e. with the expected qualitative properties), since the qualitative analysis of the models is a difficult task for non-simple systems.

   Using ordinary Petri nets, Zhou and DiCesare [47] proposed a systematic procedure for the synthesis of correct manufacturing systems. This procedure should be adapted to autonomous robotic assembly systems.

This work has reviewed the approaches that used Petri nets as a modelling methodology for assembly planning and task planning. As a conclusion, it can be stated that this is a good modelling methodology that successfully solved many problems, and that will possibly be able to cope with all the objectives and requirements posed by an autonomous robotic assembly system.

REFERENCES

1 **Gottschlich, S. , Ramos, C., and Lyon, D.** , Assembly and task planning taxonomy, *IEEE Robotics and Automation Magazine*, 1994, 4-12.

2 **Murata, T.** , Petri nets: Properties, analysis and applications, *Proc. IEEE*, 1998, 77 (0), 541-380.

3 **Silva, M.** , Practice of Petri Nets in Manufacturing, Chapman & Hall, 1993, Ch. Introducing Petri nets, pp. 1-62.

4 **Jensen, K.** , Advances in Petri Nets, Lecture Notes in Computer Science vol. 483. Springer-Verlag, 1991, Ch. Coloured Petri Nets: A High-level Language for System Design and Analysis, pp. 44-122.

5 **Jensen, K.** , Advances in Petri Nets, Lecture Notes in Computer Science vol. 483. Springer-Verlag, 1991, Ch. Predicate/Transition nets, pp. 3-43.

6 **Jensen, K.** , Advances in Petri Nets, Lecture Notes in Computer Science vol. 483. Springer-Verlag, 1991, Ch. Hierarchies in Coloured Petri Nets, pp. 215-246.

7 **Buchholz, P.** , Application and Theory of Petri Nets 1994, Lecture Notes in Computer Science vol. 815. Springer-Verlag, 1994, Ch. Hierarchical High Level Petri Nets for Complex System Analysis, pp. 119-138.

8 **Störrle, H.** , An evaluation of high-end tools for petri nets, Technical report, Institut für Informatik, Universität München (1998).

9 **Debaque, A. C., *et al*.**, The ECORP approach to petri net tool evaluation, in: Proc. of the Canadian Conf. on *Electrical and Computer Engineering*, 1994, Vol. 2, pp. 814 -820.

10 **Jensen, K.** , Overview of Design/CPN, http://www.daimi.au.dk/designCPN/over/.

11 **Zussman, E.,** and **Zhou, M.**, Design and implementation of an adaptive process planner for disassembly processes, *IEEE Trans. on Robotics and Automation*, 2000, 16 (2), 171-179.

12 **Tang, Y., Zhou, M.** and **Caudill, R. J.**, An integrated approach to disassembly planning and demanufacturing operation, in: Proc. of the IEEE Int. Symp. on *Electronics and the Environment*, 2000, pp. 354-359.

13 **Zhang, W.** , Representation of assembly and automatic robot planning by Petri net, *IEEE Trans. on Systems, Man and Cybernetics*, 1998, 19 (2), 408-422.

14 **Homem de Mello, L. S. and Sanderson, A. C.**, AND/OR graph representation of assembly plans, *IEEE Trans. on Robotics and Automation*, 1990, 6 (2), 188-199.

15 **Cao, T. and Sanderson, A. C.**, AND/OR Net representation for robotic task sequence planning, *IEEE Trans. Syst., Man and Cybernetics - Part C: Applications and Reviews*, 1998, 28 (8), 104-218.

16 **Suzuki, T. *et al.***, On algebraic and graph structural properties of assembly Petri net, in: Proc. of the IEEE Int. Conf. *on Robotics and Automation*, 1993, pp. 507-510.

17 **Cao, T. and Sanderson, A. C.**, Modelling of sensor-based task plans using fuzzy Petri nets, in: Proc. of the Fourth Int. Conf. on *Computer Integrated Manufacturing and Automation Technology*, 1994, pp. 73 -80.

18 **Zha, X.**, An integrated intelligent approach and system for rapid robotic assembly prototyping, planning and control, in: Proc. of the IEEE Int. Conf. on *Robotics and Automation*, 1999, Vol. I, pp. 108-113.

19 **Ramirez-Treviño, A. and Lopez-Mellado, E.** , Qualitative modeling of assembly tasks in robot cells using coloured nets, in: Proc. of Rensselaer's Second Int. Conf. on *Computer Integrated Manufacturing*, 1990, pp. 475-482.

20 **Groppetti, R. , Santucci, A.** and **Senin, N.** , On the application of coloured Petri nets to computer aided assembly planning, in: Proc. of the IEEE Symp. on *Emerging Technologies and Factory Automation*, 1994, pp. 381-387.

21 **Zussman, E. and Zhou, M.** , A methodology for modeling and adaptive planning of disassembly processes, *IEEE Trans. on Robotics and Automation*, 1999, 15 (1), 190-194.

22 **Moore, K. E., Gungor, A. and Gupta, S. M.**, Disassembly Petri net generation in the presence of XOR precedence relationships, in: Proc. of the IEEE Int. Conf. on *Systems, Man and Cybernetics*, 1998, Vol. I, pp. 13-18.

23 **Kasturia, E. , DiCesare, F. and Desrochers, A.** , Real time control of multilevel manufacturing systems using colored nets, in: Proc. of the IEEE Int. Conf. on *Robotics and Automation*, 1988, Vol. 2, pp. 1114 -1119.

24 **Au, Y. M. and Tam, H. Y.**, Model-based control of flexible assembly systems using colored Petri net, in: Proc. of the IEEE Int. Conf. on *Industrial Technology*, 1996, pp. 147-151.

25 **Borusan, A.** , Coloured Petri net based modelling of FMS, in: Proc. of the IEEE Int. Conf. on *Systems, Man and Cybernetics*, 1993, Vol. 1, pp. 54-59.

26 **Descotes-Genon, B.** , Coloured Petri nets: A same tool for modelling, simulation and control of manufacturing systems, in: Proc. of the IEEE Int. Conf. on *Systems, Man, and Cybernetics*, 1994, Vol. 2, pp. 1677-1682.

27 **Hsieh, F. S.**, Reconfigurable fault tolerant deadlock avoidance controller synthesis assembly production processes, in: Proc. of the IEEE Int. Conf. on *Systems, Man, and Cybernetics*, 2000, Vol. 4, pp. 3045-3050.

28 **Shiu, S. C. K.** ***et al***., Evaluation of circuit board assembly manufacturing systems using fuzzy colored Petri nets, in: Proc. of the 1998 IEEE Int. Conf. on *Systems, Man, and Cybernetics*, 1998, Vol. 2, pp. 1506-1511.

29 **Rosell, J. , Basañez, L. and Suárez, R.** , Compliant-motion planning and execution for robotic assembly, in: Proc. of the IEEE Int. Conf. on *Robotics and Automation*, 1999, pp. 2774-2779.

30 **Moseman, H. and Whal, F. M.**, Automatic decomposition of planned assembly sequences into skill primitives, *IEEE Trans. on Robotics and Automation*, 2001, 17 (5), 709-718.

31 **McCarragher, B. J.**, Petri net modelling for robotic assembly and trajectory planning, *IEEE Trans. on Industrial Electronics*, 1994, 41 (6), 631-640.

32 **Xiao, J. and Ji, X.** , A divide and merge approach to automatic generation of contact states and planning of contact motion, in: Proc. of the IEEE Int. Conf. on *Robotics and Automation*, 2000, pp. 750-756.

33 **Rosell, J.** , Local contact state space generation using colored Petri nets, in: Proc. of the 6th IEEE Int. Workshop on *Discrete Event Systems*, 2002, pp. 143-148.

34 **Simon, D. , Castaneda, E. C. and Freedman, P.** , Design and analysis of synchronization for real-time closed-loop control in robotics, *IEEE Trans. on Control Systems Technology*, 1998, 6 (4), 445-461.

35 **Suh, I. H.** ***et al.***, Design of a supervisory control system for multiple robotic systems, in: Proc. of the IEEE/RSJ Int. Conf. on *Intelligent Robots and Systems*, 1996, pp. 332-339.

36 **Lyons, D. M. and Arbib, M. A.**, A formal model of computation for sensor-based robotics, *IEEE Transactions on Robotics and Automation*, 1989, 2 (3), 280-293.

37 **Montano, L., García, F. J. and Villarroel, J. L.**, Using the time Petri net formalism for specification, validation, and code generation in robot-control applications, *The Int. Journal of Robotics Research*, 1996, 19 (0), 59-76.

38 **Mortensen, K.**, Automatic code generation from coloured Petri nets for an access control system, in: Proc. of the *CPN Workshop*, 1999 (Technical Report PB-541).

39 **Ullman, D.** , Elements of ML Programming, Prentice-Hall, 1993.

40 **Feldmann, K. *et al.***, Specification, design, and implementation of logic controllers based on colored Petri net models and the standard IEC 1131. I. Specification and design, *IEEE Trans. on Control Systems Technology*, 1999, 7 (6), 657-665.

41 **Feldmann, K**. *et al.*, Specification, design, and implementation of logic controllers based on colored Petri net models and the standard IEC 1131. II. Design and implementation, *IEEE Trans. on Control Systems Technology*, 1999, 7 (6), 666-674.

42 **Moody, J. O. and Antsaklis, P. J.**, Petri net supervisors for DES with uncontrollable and unobservable transitions, *IEEE Trans. on Automatic Control*, 2000, 45 (3), 462-476.

43 **Guéré, E. and Alami, R.** , Let's reduce the gap between task planning and motion planning, in: Proc. of the IEEE Int. Conf. on *Robotics and Automation,* 2001, pp. 15-20.

44 **Song, M. , Tarn, T.-J. and Xi, N.** , Integration of task schedulling, action planning, and control in robotic manufacturing systems, *Proc. of the IEEE*, 2000, 88 (7), 1097-1107.

45 **Cao, T. and Sanderson, A. C.**, Task decomposition and analysis of robotic assembly tasks plans using Petri nets, *IEEE Trans. on Industrial Electronics*, 1994, 41 (6), 620-630.

46 **Thomas, J. P., Nissanke, N. and Baker, K. D.**, A hierarchical Petri net framework for the representation and analysis of assembly, *IEEE Trans. on Robotics and Automation*, 1996, 12 (2), 268 - 279.

47 **Zhou, M. and DiCesare, F.** , Petri net synthesis for discrete event control of manufacturing systems, Kluwer Academic Publishers, 1993.