Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Knowledge-Intensive Teaching Assistance System for Industrial Robots Using Case-Based Reasoning and Explanation-Based Learning

Guanfeng Sun*, Tetsuo Sawaragi*, Yukio Horiguchi* and Hiroaki Nakanishi*

Graduate School of Engineering, Kyoto University, Kyoto 615-8540, Japan
(e-mail: sun.guanfeng.63m@st.kyoto-u.ac.jp,{sawaragi, horiguchi, nakanishi}@me.kyoto-u.ac.jp)

Abstract: This paper presents a novel human-system collaborative robot-programming platform, where case-based reasoning (CBR) and explanation-based learning (EBL) are integrated together. CBR takes advantage of its unique CBR cycle to achieve the knowledge acquisition and reuse in the form of case, realizing efficient robot programming with the support of experienced human experts. EBL optimizes the rule structure in the knowledge base through learning in retrieving speedup rules in order to accelerate the case adaptation process. Feasibility of this proposal is verified via a number of experiments that allow the system to output both schemata for generalized robot programming tasks whose calculation rules are adaptive enough so that it can be applied to novel task inputs. Moreover, it is shown that our system is adaptive to the increase of the cases processed and be able to tackle with the learning utility problem.

Keywords: Human-machine systems, human-centered design, interactive approaches, knowledge-based systems, and knowledge transfer, robot programming.

1. INTRODUCTION

As manufacturing shifts away from low mix/high volume production to high mix/low volume production, the current robot-programming paradigm, which is executed in a trial-and error manner, is so tedious and time-consuming to keep pace with rapidly changing customers' needs. This is mainly because a teaching task for the robots depends heavily on the engineers' experiences and knowledge and the so-called knowledge transfer issues from experienced engineers to the younger generation were not resolved at all. Then, a new robot-programming paradigm emphasizing efficiency and flexibility becomes one of the key factors for manufacturer to survive in the fierce competition.

This paper explores a new robot-programming paradigm integrating a number of machine learning methodologies with knowledge engineering, putting a high emphasis on knowledge acquisition and reuse. Knowledge to be gained from experienced engineer is in the form of cases, and the framework for the knowledge-intensive platform is designed as an extended CBR (Case-Based Reasoning) system. All the cases are organized in a hierarchical tree structure using a conceptual clustering technique that contributes much to an overall CBR framework in acquiring and reusing the knowledge via its CBR cycle. At the same time, another machine learning technique of EBL (Explanation-Based Learning) optimizes the structure of the rule base, generating speedup rule incrementally to improve case adaptation ability.

There are two main contributions of our system. The primary contribution is our emphasis on knowledge acquisition and reuse. During the process of programming, not only the solution is desired, but the knowledge that human expert use is much more preferred as well. Compared to the rule-knowledge, case-knowledge is more efficient and simple for human to learn and reuse. Besides, as an analog to human problem solving paradigm, CBR makes the incremental knowledge acquisition and reuse more natural. The secondary contribution is that our paradigm is based on learning from demonstration of the program written by experts in the past. This idea stems from the concept that text-based program is filled with experts' knowledge such as how to select point and operation strategies, discussed by Wang et al. (2012). Following this research, in this paper we focus on practical problems encountered for realizing the teaching assistance system with respect to the flexible organization of individual cases and to the improvement of case adaptation performance without suffering from the learning utility problem caused by the increase of the number of cases.

2. RELATED WORKS AND METHODOLOGIES

2.1 Related Works

In order to explore a new approach for robot programming, a great number of researches were carried out and systems were constructed. Aleotti et al. (2004) applied a virtual reality technology for robot programming. But VR needs well-structured environment, and apparently ignores the knowledge succession between the human worker and the system. Galangiu et al. (2011) constructed an expert system for robot programming. However, the establishment of expert system is time-consuming and inefficient itself, due to the fact that rules must be carefully extracted in order to avoid inconsistency. Besides, some fragments of knowledge are too ambiguous to be expressed clearly. Ushioda et al. (2006) utilized machine learning (i.e., swept volumes) for robot programming. As his method is limited to specific case and the retrieved knowledge is difficult for the system, even for a human, to interpret and reuse, it fails to satisfy low-volume and high variety manufacturing needs.

2.2 Methodologies

The robot programming is established on CBR framework, while EBL in this framework is responsible for enhancing case adaptation. As a fashionable research direction in 1980, CBR has covered many engineering fields, ranging from mechanical design to error analysis. However, there are a limited number of research works that applied CBR to the assembly domain except (Seo et al., 2007) and (Su, 2007).

CBR is selected for the following reasons. First, CBR manages to handle situations where knowledge is difficult to express clearly and completely. In practical engineering field, it is hard to express every piece of knowledge clearly in the form of rule. Besides, the process of encoding rules into the knowledge base appears rather time-consuming. Therefore, compare to rule-type knowledge, it turns out to be more appropriate to manage experience and knowledge of skilled engineer's in the form of case. Second, CBR is consistent with human cognitive model to solve problems based on similar experience, which allows the system to have strong self-learning and adaption ability in comparison with well-structured expert system and VR.

As for EBL, it is a deductive learning method that exploits a very strong domain theory to acquire knowledge from training examples (DeJong et al., 1986). EBL does not result in the acquisition of truly "new" knowledge. Instead, EBL can be viewed as a process that rearranges the knowledge structures in the domain theory to generate the operational shortcut. EBL was first adopted by Segre and DeJong (Segre et al., 1985) to give a manipulator the intelligence of constructing schema to handle some manipulation problems, while H. Adeli (Adeli et al.,1990) applied EBL in the civil engineering field to investigate structural design.

The reason why we introduce EBL into the system is two-fold. First, the EBL is able to gain the operational knowledge from even a single training example, unlike other inductive learning that needs processing a large number of samples. Considering the increasing low volume/high variety needs, this feature eliminates the time to collect different samples as production scenario varies, making it a suitable approach towards low-volume/high variety production. Second, EBL is able to generate speedup rules to accelerate the inference process and improve adaptation ability. More powerful inference does lead to efficiency improvement in case adaptation.

3. SYSTEM ARCHITECTURE

The system is composed of four components, which are Planning Part, Learning Part, Knowledge Base and Case Base, which is displayed in Fig.1. Rule base can be further divided into atomic rule base and speedup rule base. Operational criterion is also stored in the rule base. Both the planning part and the learning part interact with the case base and the rule base.

Input is supposed to be the information such as contact type from CAD model, and position and attitude of workpiece and obstacle captured by stereo camera and so on. Since how to acquire information from CAD model and sensor is not the primary focus in this research, it's assumed that a representation vector and assertions with regard to the target

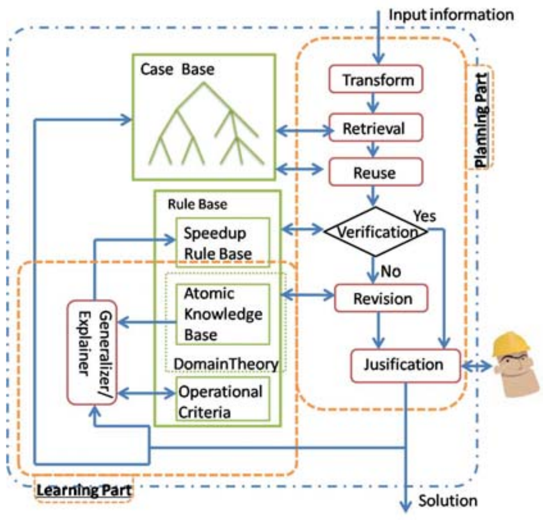


Fig.1 Proposed system architecture

case are provided as the input. On the other hand, output is a justified solution, including schema and calculation rules.

Our system is based on the case-based reasoning framework, and CBR has four processes, which are retrieve, reuse, revise and retain (Kolodner, 1992; Lopez de Mantaras et al., 2005). From a perspective of CBR cycle, the planning part undergoes three processes including RETRIEVE, REUSE and REVISE. It retrieves the most similar case (i.e., a base case) to the current problem case in question (i.e., a target case) from case base, reuses its solution and revises its solution accordingly to create a solution to the target case. RETAIN is implemented in two ways: The system managed target cases into case base hierarchically, while the EBL part acquired new speedup rules into the knowledge base to enhance the adaption ability.

3.1 The Planning Part

The planning part aims to produce a justified program based on the input. According to specific function, it can be further divided into a number of sub-parts, which are Transform, Retrieval, Reuse, Verification, Revision and Justification.

Transform part is in charge of producing a representation vector and assertions based on the information captured by sensor and CAD data. The vector is used for retrieval, as assertions are used for verification and revision.

Retrieval part takes the representation vector as an index, searching case base for the most similar base cases to target case. The indices of the matches will be output to reuse part.

Reuse part has the responsibility to acquire the solution of similar cases from the case base according to these indices given by retrieval part, and sends it to the verification and revision part. The solution is hierarchically divided into three layers, which are schema, calculation rules for major parameter and minor parameter.

Verification part verifies whether the solution matches target case or not in two aspects: schema and calculation rule. If matched, verification part outputs a solution directly to justification part. If not, verification part reports mismatch information to revision part.

Revision part revises the suggested solution based on the report from verification part and assertions. It interacts with the knowledge base for related rules to adapt the solution towards the target case, before outputting the solution for a human expert to justify.

3.2 The Learning Part

Learning part can be viewed as a retain part in the system, which is responsible for knowledge acquisition. The input to the learning part is the justified solution including program and calculation rules. After the learning part takes the input, the knowledge will be acquired in two dimensions. One is that the solution will be sent to Case base, so that it can be retained into the case base constructed in a hierarchical tree structure according to its representation vector. The other is that the calculation rules alone will be taken a goal concept for EBL to generate speedup rules. In the initial stage when there are not so many cases in the case base, experienced worker can input the vector and calculation rules directly. In the situation, the case will be sent and stored into Case base.

Learning part hosts two components, which are Explainer and Knowledge base. The atomic rules and operational criteria are stored in the knowledge base for Explainer to refer. Explainer interacts with Atomic knowledge base to construct an explanation tree. Then, Generalizer decides to what extent the derived explanation tree should be generalized for the usage of novel cases according to the operational criteria. By investigating into a derived explanation tree, speedup rule can be generated and is stored at the speedup rule base to accelerate the inference process in the future reuse. These speedup rules are treated with higher priorities over atomic rules when they are used in the further inferences.

Finally, a procedure of Justification is carried out by a human expert. Human expert is allowed to ensure the quality of solution during this stage. He/she either modifies minor flaws, or rewrites and gives the schema and calculation, if the solution is far away from satisfactory.

4. THE PLANNING MECHANISM

The following gives a detailed explanation on the planning mechanism of our system. We talk about how to represent a case in a representation vector, how the system retrieves similar cases from the case base and how the system reuses and revises them.

4.1 Case Representation and Organization

Here are some assumptions set here to simplify this research. Both workpieces and obstacles are restricted to be kinds of concrete cuboid. Besides, both initial and destination environment is only allowed to have one obstacle or not. The tool is set two-figure gripper. The task is pick-and-place operation. Information such as attitude and position acquired by a visual sensor and contact type is transformed into a vector to represent the case.

Herein, the five attributes are used to constitute a vector; 1) obstacle in initial environment, 2) obstacle in destination environment, 3) type of initial/destination environment combination, 4) contact type, and 5) attitude relation of the workpiece between the initial and the destination state.

The initial environment and the destination environment are assumed to be limited into either of the three types that are labeled by taking account of the relations among workpieces, obstacles and tools; either of “isolated”, “collision-free”, “collision-prone”. The attribute concerning with the type of initial/destination environment combination considers obstacles in both initial and destination environment at the same time. The values for this attribute are defined as the combination of the above three values, thus may take one of the nine values. The values for the attribute of the contact types are either of; “plane-on”, “loose-insert” and “tight-insert” that is determined according to the adopted strategy to place the workpiece. The values for the attribute of relation between initial state and destination states of workpiece include “unchanged”, “lean” and “reverse”, that are affecting on deciding PickPoint parameter in the robot programming.

4.2 Case Retrieval

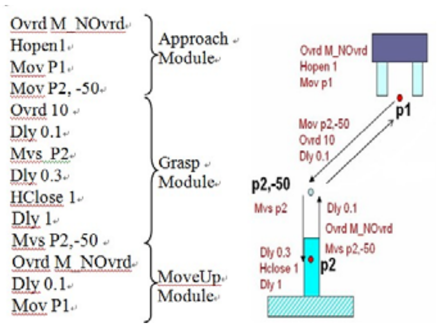
In our system, a method of conceptual clustering called as COBWEB (Fisher, 1987) is adopted for organizing the cases so that the base case could be retrieved efficiently in reply to the input novel case. Here are the two primary motivations to use the concept clustering. First, the category utility, which is used as a heuristic metric controlling the organization of the clusters, is consistent with human’s unsupervised categorization. It is based on probability matching, which is a cognition phenomenon, allowing clustering not to be susceptible to irrelevant and incomplete concepts. Second, since our system is to be expanded according to the accumulation of incrementally provided cases, a capability of incremental learning is essential, and during this incremental learning, the search cost should be favourable avoiding exhaustive search methods. Low computation cost of conceptual clustering permits system to update case base rapidly when each new case is added, thus sustaining a continual basis for reacting to new provision of inputs incrementally. In this work, as an initial set of case base, 61 different cases are generated, whose values for the corresponding attributes are picked up randomly from their corresponding domain values. Then, conceptual clustering is applied to these cases, wherein category utility (Corter and Gluck, 1992) is used to guide clustering in measuring both similarity of objects within the same class and dissimilarity of objects from the ones of different classes.

4.3 Deeper Case Description

After the system retrieves the similar cases from the case base, it will reuse the knowledge of the retrieved cases to create a solution to the target case. As mentioned before, the knowledge attached to the previous case (i.e. case knowledge) is divided into three layers, which are schema, major parameter and minor parameter. It is analogous to human’s process toward robot programming, which is that expert defines the schema by writing the program in controller-specific language, then determines the parameter by teaching robot the information concerning parameters via teaching pendent.

A. Schema Level

A schema is the program in controller-specific language, which provides strategies to be applied for the target case. Figure 2 shows a simple program written for the industrial robot MELFA BFP-A86466 made by Mitsubishi Electric Corporation (Mitsubishi, 2005). The program refers to a task to pick up a cuboid from the plane. The whole motion is to move to SafePoint (p1) at first, then move to PrePickPoint (p2,-50), translate to PickPoint (p2) to grasp the object and retreat. Ovrd is the command that means changing the velocity, as M_NOvrd and 10 stands for the maximum velocity and 10% of maximum velocity, respectively. Mov features move, while Mvs represents translate. The point following the Mov and Mvs shows the target point. Dly means delay, and the following number shows the time delay in seconds.



```
Ovrd M_NOvrd
Hopen1
Mov P1
Mov P2, -50
Ovrd 10
Dly 0.1
Mvs  P2
Dly 0.3
HClose 1
Dly 1
Mvs P2,-50
Ovrd M_NOvrd
Dly 0.1
Mov P1
```

Fig.2 Pick Operation

Due to flexibility and inter-changeability, a schema is further divided into several segments. Its hierarchical task structure is depicted in Fig. 3.

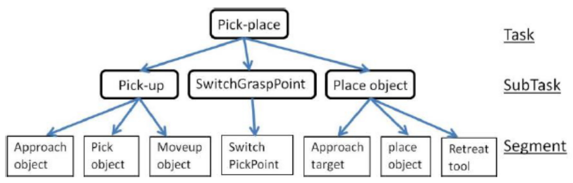


Fig.3 Hierarchical Representation of Pick-Place Task

B. Rules for Calculating Major and Minor Parameters

Major parameters are defined as key parameters that are directly relevant with and contribute to the success of the assembly. The calculation rule is used in this research to express the knowledge of major parameter. Along with specific data such as workpiece geometry, the detailed parameter can be calculated. In the simple task above, the major parameters are PickPoint(p2), and PrePickPoint(p2, -50). Inappropriate selection of major parameters may result in the failure of the assembly task. Compared to major parameters, minor parameters do not define the success of assembly, but affect the quality of the assembly task performance. Two important minor parameters are known to be velocity and time delay.

The velocity changes frequently during the operations and is closely related both with the stability and the efficiency of the assembly task, while the time delay set before and after the gripper's actions should be also selected appropriately by considering the trade-off between stability and efficiency. The selection of minor parameter depends heavily on the experience of skilled engineers, and minor parameters will be preserved originally for human expert to justify.

4.4 Case Revision (Case Adaptation)

The prior solution of retrieved case (i.e., the base case) does not ensure to match the target case perfectly. Verification and revision is usually carried out to verify whether the solution stored in the base case is applicable to the target case or not, and to revise it if necessary. As the solution of the base case composes schema and parameters, verification and revision will be carried out in corresponding two stages.

A. Schema Verification and Revision

Schema verification is based on verifying difference pattern, which means that the system figures out what is different between the base case and target case. As the case is described by representation vector, the system works out the difference pattern by analyzing the difference between vectors and consulting the knowledge base. If there is no difference between vectors, no revision needs to be made. According to the verification results, revision part takes it to consult the knowledge base. There are a number of related rules with regard to schema revision in the knowledge base. In this manner, the system realizes the revision on schema of the retrieved case by switching related segment.

B. Calculation Rule's Verification and Revision

Calculation rule is verified by examining whether it fits with assertions or not. Calculation rule is regressed into its constraints by consulting the rules in the knowledge base. Subsequently, whether assertions can satisfy all the constraints or not is investigated. If not, calculation rule will be considered to be inappropriate to the current target case and the system will resort to the revision part to revise by sending the type of the rule. Revision part reasons in a forward-chaining fashion from the assertions by referring to the rules in the knowledge base, until it gets the related rule. If there is not related rule that can be found in the knowledge base, revision part can leave this problem to human expert in the justification stage, waiting for human expert to provide the calculation rule.

4.5 Justification

When a program is generated, there might occur some inappropriateness that is beyond the system's ability to realize. A human expert is expected to justify the program generated by the system to ensure the solution quality. Human expert either modifies slightly towards the minor problem, or rewrites the schema and redefines the calculation rules, given the suggested solution is far away from satisfactory.

A number experiments are conducted to demonstrate the planning function of the system, but the details of the demonstration are skipped because of the space limitation here. This will be shown in the presentation according to the process mentioned in this section.

5. THE LEARNING MECHANISM

The learning part handles both the case knowledge and the rule knowledge. As to the case knowledge, conceptual clustering can incorporate a case into clustering tree in the case base automatically.

As for the rule knowledge, EBL is used to create new speedup rule based on the existed rules. By using speedup rule rather than inference on a number of atomic rules step by step, the system can enhance its adaption ability, which will be discussed in the later section. Whether EBL part needs to be provoked or not depends on whether there are related speedup rules in the knowledge base or not. If related speedup rules are found in the knowledge base, this particular goal concept has got regressed before and there is no new knowledge to be learned. In this case, EBL will not be activated. If not, EBL part is to be activated to learn the speedup rule.

An example is used to demonstrate how EBL functions. At the start, the calculation rule is taken as a goal concept. Then, explainer searches within atomic knowledge base for related atomic rules to construct the explanation tree.

1, Pickpoint is -y, 0.5 if
Feasible is -y-axis, 0.5, and
Relation is reverse, and
Approachable is –y-axis.
2, Feasible is -y-axis, 0.5 if
OpenWidth bigger-than dx, and
Length bigger-than 0.5*dy, and
3, Approachable –y-axis if
-y-axis hasnot obstacle, and
x-axis hasnot Primary-obstacle
4, x-axis hasnot Primary-obstacle if
+x-axis hasnot Primary-obstacle, and
-x-axis hasnot Primary-obstacle

At this time, the generalizer checks if-parts of these rules to see whether they meet operationality criteria or not. If they do, the explainer will cease searching related rules. In this example, operational criterions used are listed as follows:

1, Relation is reverse, and
2, OpenWidth bigger-than dx, and
3, Length bigger-than 0.5*dy, and
4, -y-axis hasnot obstacle, and
5, +x-axis hasnot Primary-obstacle, and
6, -x-axis hasnot Primary-obstacle

Accordingly, an explanation tree is constructed as showed in Fig. 4.

By building a new explanatory structure successfully, a new speedup rule is created.

PickPoint is -y,0.5 if
Relation is reverse, and
OpenWidth bigger-than dx, and
Length bigger-than 0.5*dy, and
-y-axis hasnot obstacle, and
+x-axis hasnot Primary-obstacle, and
-x-axis hasnot Primary-obstacle

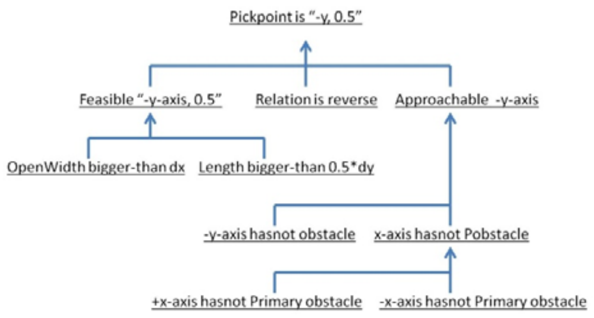


Fig. 4 Explanation tree for the rule (Pickpoint is -y, 0.5)

It means if the gripper's open width is bigger than dx, and the figure length is bigger than 0.5*dy, and the attitude relation between the initial and destination state is reverse, no obstacle is at the negative side of y–axis and no primary obstacle is on the x-axis, then the gripper can pick up the workpiece from the negative side of x-axis and its PickPoint is in the middle of workpiece. The preceding speedup rule will be stored into knowledge base and used in verification and revision in the future.

6. LEARNING UTILITY PROBLEM

In this section, we discuss whether EBL can have a positive influence on the case adaptation. First, we measure the elapse time used by the system with and without EBL function to evaluate adaptation efficiency.

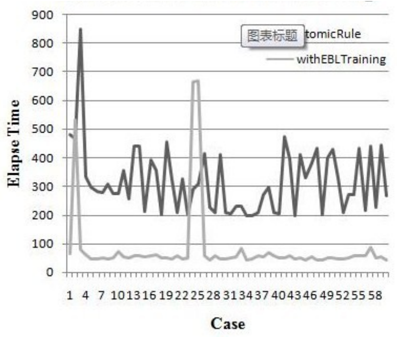


Fig. 5 Elapse time for processing cases

The experiment is set to require the system to figure out how to calculate PickPoint. The input into the system is the assertions of specific cases, while the output is calculation rules. The system without EBL obtains calculation method by making inference on those atomic rules iteratively. In contrast, the system with EBL cannot only refer to atomic rules, but retrieves and reuses the speedup rules as well.

The 60 target cases and 88 atomic rules in knowledge base are used for demonstration. During the experiment, there are roughly ten speedup rules retrieved.

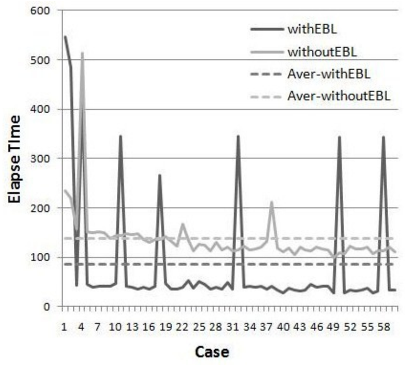


Fig. 6 Elapse Time for system via atomic rule and EBL-training

The result is displayed in Fig.5, in which x-axis stands for a list of cases and y-axis shows elapse time in microsecond for processing a case. The curve in black and grey represents the system with EBL and that without EBL. As showed in Fig. 6, although the system with EBL spends relatively longer elapse time at some points due to EBL's learning of speedup rule, its average elapse time is almost half of that used by the system without EBL. Therefore, EBL is considered to boost case adaptation by 50%. Second, ordering Effect is examined. Since the speedup rules generated previously can accelerate the adaption while faced with coming similar situations, it comes necessary to investigate whether the order of input case has something to do with case adaptation. Third, we evaluate EBL's influence by taking its contribution to solving utility problem into account. In the machine learning community, utility problem means the trade-off between solution quality and system's efficiency: large knowledge base enhances system's adaption ability and improves solution quality, while the efficiency is deteriorated with enlargement of knowledge base.

Utility problem is investigated here in two dimensions. First is by compare the performance of the system without EBL and that via EBL training. Given all 60 cases at hand, the former 20 cases are used for training, then all 60 cases are used to test whether there is utility problem or not. As Fig.6 shows, the average time for the system with EBL training is 317μs, while the system without EBL takes 823 μs. Hence, EBL training gave a 260% improvement in case adaptation. Secondary, we also evaluate utility problem in the perspective of performance response and verified that EBL helps the adaptation ability of the system improve increasingly better, telling that the EBL here does not suffer from utility problem.

7. CONCLUSIONS

A knowledge platform integrating CBR and EBL for robot programming is proposed in this paper. The applicability is verified by applying it to experiments. CBR manages to generate a solution by referring to the previous case and acquires the case knowledge. EBL is responsible for learning speedup rules to optimize knowledge base's structure to improve the system's adaption ability. Since the increase of the learned rules are assumed to deteriorate the performance of adaptation, we discussed about this learning utility problem and verified that EBL helps the adaptation ability of the system improve increasingly better without being suffered from the utility problem. How to allow EBL to perform a greater role will be investigated in the future. One idea is to permit EBL to participate in the retrieval, as the explanation tree can be used as the resource to generate new attributes. Whether retrieval quality can benefit from these new attributes needs to be investigated. If verified, EBL is able to integrate with CBR better to achieve the comprehensive improvement of system.

REFERENCES

Adeli, H. and Yeh, C. (1990). "Explanation-based machine learning in engineering design", *Engineering Applications of Artificial Intelligence*, **3**, 2, pp. 127–137.

Aleotti, J., Caselli, S. and Reggiani, M. (2004). "Leveraging on a virtual environment for robot programming by demonstration", *Robotics and Autonomous Systems*, **47**, 2–3, pp. 153–161.

Corter, J.E. and Gluck, M.A. (1992). "Explaining basic categories: feature predictablility and information", *Psychological Bulletin*, **111**, pp. 291-303.

DeJong, G. and Mooney, R. (1986). "Explanation-based learning: an alternative view", *Machine Learning*, **1**, 2, pp 145-176.

Fisher, D.H. (1987). "Knowledge acquisition via incremental conceptual clustering", *Machine Learning*, **2**, 2, pp. 139-172.

Galangiu, G.A, Stoica, M., Sarkany, I. and Sisak, F. (2011). "Expert system for teaching robots in a flexible manufacturing line", *Proc. of 15th Int. Conf. on Intelligent Engineering System*, pp. 253-257.

Kolodner, J.L. (1992). "An introduction to case based reasoning", *Artificial Intelligence*, **6**, 1, pp. 3-34.

Lopez de Mantaras, R. et al. (2005). "Retrieval, reuse, revision and retention in case-based reasoning," *The Knowledge Engineering Review*, **20**, 3, pp. 215-240.

Mitsubishi. (2005). Mitsubishi Electric Corp. Mitsubishi Industrial Robot Instruction Manual - detailed explanations of functions and operations, Art. No. 132315, Version K, BFP-A5992, Mitsubishi Electronics Corporation, 14.

Seo, Y., Sheen, D. and Kim, T. (2007). "Block assembly planning in shipbuilding using case-base reasoning", *Expert System with Application*, **32**, 1, pp. 245–253.

Segre, A.M. and DeJong, G. (1985). "Explanation-based manipulator learning: acquisition of planning ability through observation", *Proceeding of IEEE Int. Conf. on Robotics and automation*, 2, pp. 555-560.

Su, Q. (2007). "Applying case-based reasoning in assembly sequence planning", *International Journal of Production Research*, **45**, 1, pp. 29-47.

Ushioda, T., Maeda, Y. and Akiba, D. (2006). "A robot teaching method utilizing swept volumes by direct teaching", *Proc. of JSME Conf. on Robotics and Mechatronics*, 2P1-D33 (in Japanese).

Wang, L., Tien, Y. and Sawaragi, T. (2011). "Case-based automatic programming in robotic assembly production", *Industrial Robot: An International Journal*, **38**, 1, pp. 86-96.