지식베이스시스템을 적용한 휴머로이드 로봇의 자율 제스쳐 생성기법

Autonomous Gesture Generation in Humanoid Robot using Knowledge-Based System

***#정유철 [1], Laxmisha Rai [1], 홍지만 [2], 한헌수 [2]**

*#Yu Chul Jung(jycrobot@ssu.ac.kr) [1], Laxmisha Rai[1] , Jiman Hong [2], Hernsoo Hahn[2]

[1] 숭실대학교 지능형로봇연구소, [2]숭실대학교 정보통신전자공학부

Key words : Humanoid Robot, Synchronization, Knowledge-Based System, Intelligent, Component.

1. INTRODUCTION

Gesture generation in humanoid robots contribute to the overall robot behavior and further increase the abilities of human-robot interaction. Most of the humanoid gestures are synchronized coherently to generate one purposeful behavior. For example, the gesture formed with two-hands is more powerful than single hand. Two legs synchronize each other to generate walking behavior. This is same considering the movements of the hands, where they also synchronize each other while walking. The other coordinated actions such as hugging, weight lifting, picking a ball and clapping hands need proper synchronization and coordination from arms and hands. Generating monotonous gestures isn't hard for engineers, but definitely generating coordinated gestures using both arms is a challenging task. For example, training a robot to pick-up a flying ball needs coordinated and synchronized efforts from both hands along with meeting real-time constraints. However, to determine the predictability of gestures, we also need the effective tool to support all these behaviors simultaneously. In this paper we demonstrate the advantages of knowledge-based tools in generating different human gestures in humanoid-like robots.

2. PROPOSED KNOWLEDGE-BASED ARCHITECTURE

The proposed architecture is as shown in the Fig. 1. Firstly, the robot's physical components which contribute to the overall gesture generation are recognized. After identifying the components such as neck, arms, hands, the different actions possible in these components are identified. These action sequences are mapped to different *facts* in a knowledge-based system such as JESS [1]. Using these *facts*, the different *rules* are written to generate purposeful gestures. Table 1 shows the list of *facts* and added to the JESS along with parameters and descriptions. We can add new functions to the Jess language simply by writing a class that implements the *jess.Userfunction* interface , creating a single instance of this class and installing it into a *jess.Rete* object using *Rete.addUserfunction()*.

3. EXPERIMENTAL EVALUATION

Experiments are carried out with the RoMAN robot developed by the NT Research Inc in Korea [2]. RoMAN has a human-like torso on a four-wheel robot. It has a neck, and two six degree-of-freedom arms as the major physical modules. The JESS is installed in a Windows-XP based PC. Fig. 2, shows an execution of a *fact (Hand_Close 2),* which *close* the LEFT hand (all fingers are clamped inside).

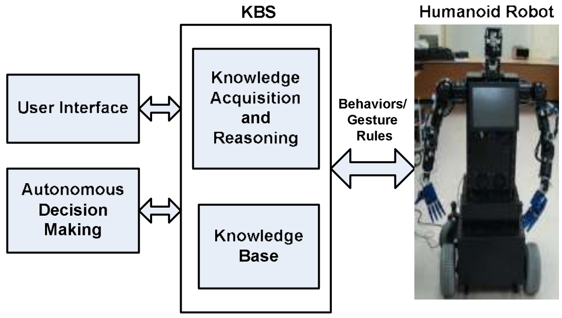

Fig. 1 Intelligent gesture generation in Humanoid robot using knowledge-based system

Table 1: List of different component specific *facts*.

fact Names	Parameters	Description
Arm_ Move	AN,X, Y, Z	Move the position of RIGHT/LEFT arm with X, Y, and Z rectangular coordinates. (AN=1 for RIGHT, 2 for LEFT)
Arm_ Joint	AN, θ_1, θ_2, θ_3, θ_4, θ_5	Move the 5 joints of the RIGHT/LEFT arm with mentioned angles (degrees). (AN=1 for RIGHT, 2 for LEFT)
Arm_ Homing	AN,JI	Arm homing based on joint index (JI) of the RIGHT/LEFT arm (AN=1 for RIGHT, 2 for LEFT) (JI range, 1~5).
Arm_ Positioning	AN	RIGHT/LEFT arm home positioning (AN=1 for RIGHT, 2 for LEFT).
Hand_ Move	AN,SP	RIGHT/LEFT hand and wrist control (AN=1 for RIGHT, 2 for LEFT).
Hand_ Open	AN	RIGHT/LEFT hand open. (AN=1 for RIGHT, 2 for LEFT)
Hand_ Close	AN	RIGHT/LEFT hand closing (AN=1 for RIGHT, 2 for LEFT).
Neck_ Home	-	Neck home positioning.
Neck_ Move	N, SP	Pan-Tilt move control (N, 1-Tilt move, 2- Pan move, and SP, step pulse, 0~1023).
Neck_Pan	N	Pan move control.
Neck_Tilt	N	Tilt move control.

Fig. 3, Fig. 4, and Fig. 5 shows the other synchronized actions generated using single *fact.* However, intelligent gesture generation include multiple *rules*, which in turn include several *facts*. Fig. 6, shows a example *rule* which guides the robot to perform several actions. The robot first bends the neck pan and then move both hands and finally lift up the neck pan. This kind of sequence is useful especially greeting the guests in different cultures.

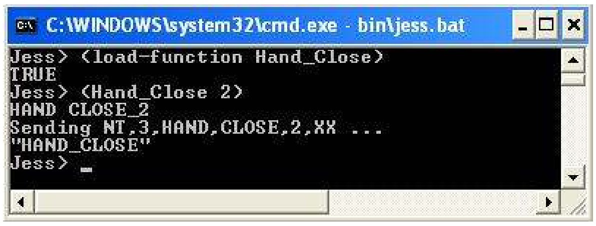

Fig. 2 Execution of *Hand_Close fact*.

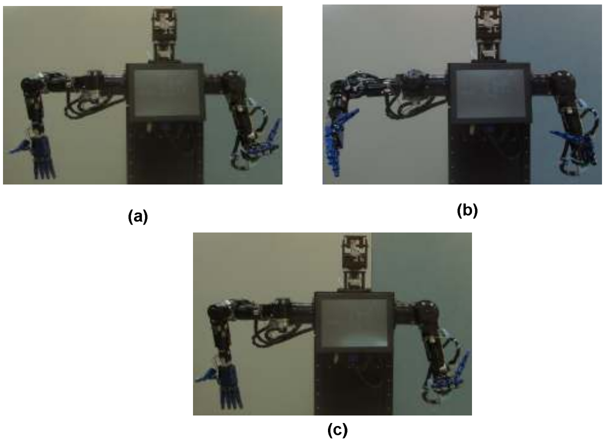

Fig. 3 The execution of *facts* (a) Initial gesture, (b) after execution of *Arm_Joint* , and (c)*Arm_Move*.

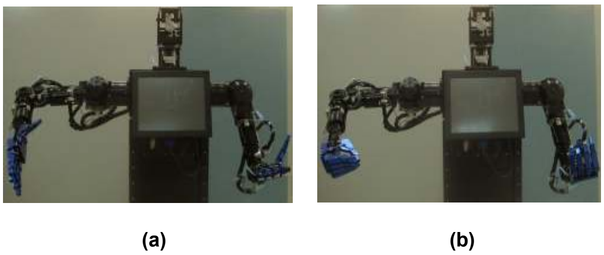

Fig. 4 (a) *Hand_Open* (b) *Hand_Close facts* operations applied to both hands simultaneously.

In some situations, various components may be assigned higher priority than other modules. For example, usually right hand has more priority than left hand. We can also set priorities to execute different actions synchronously. Each rule has a property called salience that is a kind of rule priority. Declaring a low salience value for a rule makes it fire after all other rules of higher salience. In the proposed method, user can define or edit the rules for multiple gesture generation, interpretation, and can define robot behavior according to user decisions. This will also help the user to understand the physical and behavioral constraints of the robot. Moreover, by generating different behaviors user will get close interaction with the robot.

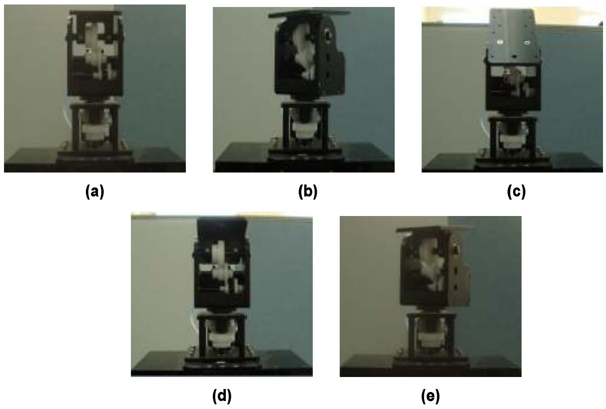

Fig. 5 *(a) Neck_Home* (b) *(Neck_Pan 512),* (c) *(Neck_Tilt -512)* ,(d) *(Neck_Move 1 500),* and (e) *(Neck_Move 2 500)*

```
(defrule rule-1
        (beh greeting)
        =>
        (load-function Arm_Move)
        (load-function Neck_Home)
        (load-function Neck_Tilt)
        (printout t "Hello Greeting," crlf)
        (Neck_Tilt -512)  (Arm_Move 1 200 400 100)
        (Arm_Move 2 200 400 100) (Neck_Home)
)
```

Fig. 6 *Rule* for generating greeting gesture

4. CONCLUSION

In the present study we implemented various mechanisms of coordinated and simultaneous operations of humanoid robot. The priority based rules are also written to test the overall behavior of the robot. Our future goal is to generate autonomous actions based on the sensor response. Knowledge-based system can effectively support to identify the robot uncertainties. A user can generate any sequence of hand movements with minimum knowledge. This also provides customer edge on understanding the robot operations. The users can easily estimate the behavior of the robot by executing few simple rules.

ACKNOWLEDGEMENT

This research was supported by the MKE(the Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute for Information Technology Advancement). (IITA-2008-C1090-0803-0006)

REFERENCES

[1] JESS, the Rule Engine for the JavaTM Platform, http://herzberg.ca.sandia.gov/jess/
[2] NT Research, Inc, www.ntresearch.net