How to use a cognitive architecture for a dynamic person model with a social robot in human collaboration

Thomas Sievers*, Nele Russwinkel

Institute of Information Systems, University of Lübeck, Ratzeburger Allee 160, 23562 Lübeck, Germany

Abstract

The use of cognitive architectures is promising in order to achieve more human-like reactions and behavior in social robots. For example, ACT-R can be used to create a dynamic cognitive person model of a human cooperation partner of the robot. A proof-of-concept for a direct and easy-to-implement integration of ACT-R with the humanoid social robot Pepper is described in this work. An exemplary setup of the system consisting of cognitive architecture and robot application and the type of connection between ACT-R and the robot is explained. Furthermore, an idea is outlined of how the cognitive person model of the human cooperation partner in ACT-R is updated with dynamic data from the real world using the example of emotion recognition by the robot.

Keywords

ACT-R, cognitive architecture, human-robot interaction, social robotics

1. Introduction

The development of situated human-aware agents that interact with human partners is a new field of research in terms of using a cognitive architecture for controlling the application and modeling human-like interaction. The use of cognitive architectures is promising in order to achieve more human-like reactions and behavior in social robots. Adaptability to changing situations in human-robot dialog and the comprehensibility and thus the acceptance of robots, even in environments that are sensitive and anxiety-inducing for humans, could also be improved as a result. This work attempts to make a first step towards the utilization of different cognitive concepts (e.g. situation understanding, prediction and adaptation to the emotional state of the partner, flexible task anticipation) by describing a proof-of-concept for the integration of a cognitive architecture with the humanoid social robot Pepper and preparing a technical basis for a more human-like perception of human interaction partners. In this context, we have carried out an initial study with the application scenario of a public authority [1]. However, a detailed evaluation and further studies that could confirm an effective benefit are still pending.

Cognitive architectures refer both to a theory about the structure of the human mind and to a computational realization of such a theory. Their formalized models can be used to further refine a comprehensive theory of cognition in order to provide common ground for working towards a specific goal, and to flexibly react to actions of the human collaboration partner and to develop situation understanding for adequate reactions. Well-known and successfully used cognitive architectures are ACT-R (Adaptive Control of Thought - Rational) and SOAR [2].

Like any cognitive architecture, ACT-R as a theory for simulating and understanding human cognition aims to define the basic and irreducible cognitive and perceptual operations that enable the human mind. In theory, each task that humans can perform should consist of a series of these discrete operations. Most of ACT-R's basic assumptions are also inspired by the progress of cognitive neuroscience, and ACT-R can be seen and described as a way of specifying how the brain itself is organized to produce cognition [3].

For an envisioned scenario, this cognitive architecture can generate flexible task knowledge and build mental representations of the relevant information about the individual with whom the robot is collaborating, the state of the task to be accomplished together and/or the person model of the human. If at some point it turns out that the intention of the human cooperation partner cannot be achieved directly because, for example, some relevant information is missing, this person will probably be frustrated. When something fails in completing the desired task, the human perception of the robot can be a critical component for the acceptance of social robots in general. Greater autonomy of the robot can lead to greater blame if something goes wrong. In their workshop report, Förster et al. provide a comprehensive overview of all the things that can go wrong in conversations between humans and robots, including a detailed analysis of failures [4]. Appropriate reactions need to be retrieved by the robot to relate to possible failures, e.g. to find an alternative solution. Frustration on the part of the human counterpart should be avoided as far as possible. [5].

After giving some examples from previous research on connections between ACT-R and robots, we present our exemplary system setup, which consists of the cognitive architecture and a robot application programmed for the purpose of a direct connection between ACT-R and the robot. The *standalone application* of ACT-R we use is available for the main computer platforms Linux, macOS and Windows. We show a dynamic update of the cognitive person model of the human cooperation partner in ACT-R with data from the real world using the example of emotion recognition by the robot.

2. Related work

A coupling of ACT-R as a cognitive architecture with different types of robots has already been realized and used for various purposes. For example, an interactive narrative system is described in which the characters in the story are interpreted by humanoid robots, which is achieved by defining suitable cognitive models [6]. These robots are using the NarRob framework [7].

A storytelling robot controlled by ACT-R is able to adopt different persuasion techniques and ethical stances while

Workshop Robots for Humans 2024, Advanced Visual Interfaces, Arenzano, June 3rd, 2024

*Corresponding author.

sievers@uni-luebeck.de (T. Sievers); russwinkel@uni-luebeck.de (N. Russwinkel)

0000-0002-8675-0122 (T. Sievers); 0000-0003-2606-9690 (N. Russwinkel)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

talking about certain topics [8]. In this case, the cognitive ACT-R architecture is connected to a Unity 3D engine.

An adaption of the ACT-R architecture for embodiment, then called Adaptive Character of Thought-Rational/Embodied (ACT-R/E) was created to function in the embodied world, placing an additional constraint on cognition namely that cognition occurs within a physical body that must navigate in real surroundings, as well as perceive the world and manipulate objects [9].

ACT-R is also used in human-robot collaboration (HRC) for mobile service robots, connecting and integrating modules of human, robot, perception, HRI, and HRC in the ACT-R architecture [10].

The inner voice of a robot cooperating with human partners is made audible via ACT-R integrated in the Robot Operating System (ROS) [11]. Also an implementation of a robotic self-recognition method by inner speech is demonstrated by using ACT-R [12].

The distinctive feature of this approach is that the robot is directly connected to the ACT-R environment via Wi-Fi without using a special framework. It is therefore not necessary to install the ACT-R application on the robot in order to run the model. In this way, there is no need to deal with specific requirements of a particular framework.

3. Connect ACT-R to a Pepper robot

The ability of ACT-R as a system to perform a wide range of human cognitive tasks can be directly combined with a social robot that interacts with humans. The assumption behind these efforts is that this could make a conversation between a robot and a human more human-like on the part of the robot and thus more pleasant for the human.

3.1. ACT-R

The basic mechanism of ACT-R consists of the main components *modules*, *buffers* and *pattern matcher* [13]. There are two types of modules: Perceptual-motor modules forming the interface with the real world (*motor module* and *visual module*), and the memory modules comprising *declarative memory* consisting of facts and *procedural memory* consisting of productions. Productions represent knowledge about how something should be done. Figure 1 gives an overview of the main components.

ACT-R accesses its modules (with the exception of the procedural memory) via special buffers. The buffers form the interface to this module. The buffer content represents the state of ACT-R over time. The pattern matcher attempts to find a production that corresponds to the current state of the buffers. Only one production can be executed at a time. Productions can modify the buffers during execution and thus change the state of the system. Cognition is therefore represented in ACT-R as a sequence of production firings.

In our approach, we do not use the visual and motor modules to provide input to the system. The buffers are used directly to exchange information between the real world of the robot and the ACT-R model.

3.2. Humanoid robot Pepper

The social humanoid robot Pepper [14] as seen in Figure 2 developed by Aldebaran is 120 centimeters tall and optimized for human interaction. It is able to engage with people through conversation, gestures and its touch screen. Pepper can focus on, identify, and recognize people. Speech recognition and dialog is available in 15 languages. Beyond, Pepper manages to perceive basic human emotions. The robot features an open and fully programmable platform so that developers can program their own applications to run on Pepper.

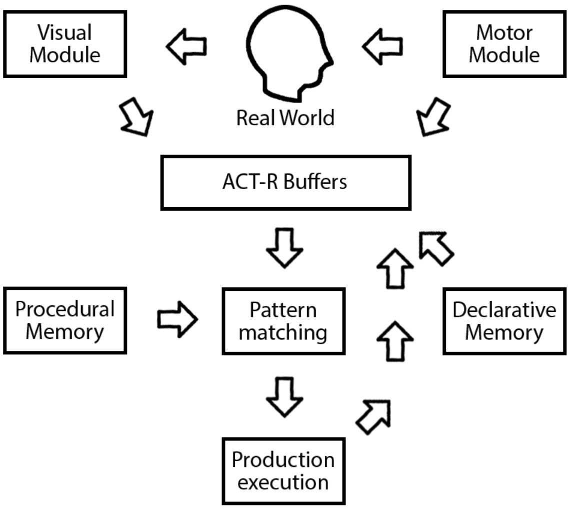

Figure 1: ACT-R modules, buffers and pattern matcher

Since research has generally shown that trust is the basis for successful communication tasks and trust in robots is increased by anthropomorphism, a humanoid social robot like Pepper is a good choice for social interaction and the provision of services when dealing with customers. A human face, the possibility of human-like expressions and body language and the use of voice are seen as beneficial for the trust of customers in the robot [15]. It has the advantage over a chatbot that it also shows physical gestures, which makes communication much more vivid and strengthens a personal relationship. The Pepper robot is already being used in many HRI projects and has also been tested in real production use.

3.3. The robot application

We developed an application that controls the robot's reactions to what the human conversation partner says. To do this, we used Android Studio with the Kotlin programming language and the Pepper SDK for Android [16], which enables the robot to be controlled via an app from its Android tablet. The Pepper SDK as an Android Studio plug-in provides a set of graphical tools and a Java respectively Kotlin library, the QiSDK, so that specific functionalities of Pepper's operating system could be used in a straightforward way directly from an Android application, e.g., for focusing upon a person, listening, talking and chatting as well as movements of head and arm to stress what has been said.

3.3.1. Listen and talk

Pepper's native speech recognition capabilities and a speech output with the – in our case German – language pack are used for speech input and output and Pepper's *Chat feature* [17] is utilized to conduct the dialog. The chat feature

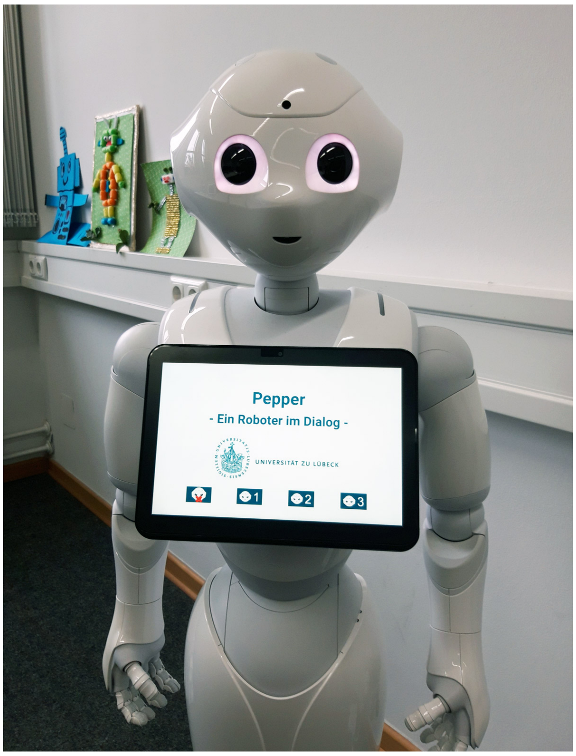

Figure 2: Humanoid Robot Pepper

allows the robot to understand individual words and short phrases even if they are spoken as part of a longer sentence. Words and phrases that the robot should understand, as well as the corresponding answers, are stored in topic files in the form of dictionaries and dialog branches. The flexible options for using variables or randomly selected parts of sentences in the robot's responses enable a natural dialog flow. The Pepper SDK also provides parameters for using pauses, intonation and voice modulation to further enhance a human-like dialog.

With regard to controlling the reactions and statements of the robot by an ACT-R model, which is supplied with relevant data for interaction from the real world, the use of these topic files offers for the robot the possibility to make statements adapted to the current situation by referring to the appropriate sections in the topic file. Figure 3 shows a schematic diagram of the topic file process within the robot application.

3.3.2. Animation

Robot gesture animation depending on a specific context can be used to support what is said depending on the situation. These animations increase anthropomorphism and comprehensibility through the indirect effect of body language. Groups of suitable animations can be defined, of which a randomly selected one is executed at certain points of the interaction, e.g. when greeting, in response to a question from the human, when the robot asks a question, etc. These animations support the interaction with the human as they emphasize the robot's statements.

Depending on the course of the conversation and the findings about the emotional state of the human counterpart, for example, the ACT-R model can be used to control the robot's gestures in conjunction with the robot's utterances.

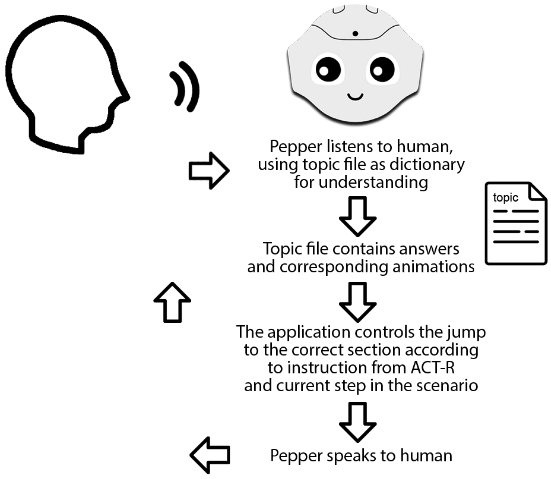
Pepper listens to human, using topic file as dictionary for understanding
topic
Topic file contains answers and corresponding animations
The application controls the jump to the correct section according to instruction from ACT-R and current step in the scenario
Pepper speaks to human

Figure 3: Schematic diagram of the topic file process

3.4. System setup for ACT-R and the robot

The *standalone* version of ACT-R is used for this work, i.e. the application provided at https://act-r.psy.cmu.edu/ instead of running the Lisp sources. To establish a remote connection from the robot to ACT-R, the remote interface – the *dispatcher* – has to be used, which is implemented by a central command server. The ACT-R core software connects to this dispatcher to provide access to its commands, and the dispatcher accepts TCP/IP socket connections that allow clients to access these commands and provide their own commands for use. The commands available via the dispatcher can be used wherever a Lisp function was formerly required. By default, the standalone version forces the dispatcher to use the *localhost* IP address of the computer on which it is running for connections instead of an external IP address. This means that only programs on the same computer can establish a connection, and once ACT-R has been started, this can no longer be changed. To disable this function, the file force-local.lisp must be removed from the ACT-R/patches directory before the application is executed. Then it will use the machine's real IP address for the dispatcher's connections and setting **allow-external-connections** in the model file will let other machines connect. Another option is to place the model file in the ACT-R/user-loads directory. External connections are then always permitted. The address and port used by the dispatcher is displayed at the top of the ACT-R terminal window. This information must be used on the remote computer for connection.

The Pepper application contains a program section for the remote connection to the dispatcher. This client connection can be used to start and control an ACT-R model that maps the cognitive processes for controlling human-robot interaction. The client is able to interact directly with the model by calling commands. The *run-full-time* command, for example, together with a number of seconds, starts and runs the model for the specified time. The *evaluate* method is used to evaluate commands from the dispatcher. It requires the name of the command to evaluate.

3.4.1. The ACT-R model

The ACT-R model created in Lisp for this proof-of-concept study uses a goal slot *pepper_out* for sending commands to the client application using ACT-R productions. This goal slot is evaluated via a permanently running while loop using the *buffer-slot-value* command that gets the value of a slot from the chunk in a buffer of the current model. The *buffer-slot-value* is sent as a string in JSON format via the TCP/IP socket stream. Each evaluation command is assigned a unique ID. This ID is used to identify the correct part of the data in the stream received by the socket. The permanent evaluation of the content of the goal slot *pepper_out* in the client application is used to create special commands for the robot depending on this slot content, e.g. to execute a certain animation or to make a corresponding utterance.

To illustrate the syntax, the following lines show an example of using the *evaluate* method for the retrieval of a goal slot as a control signal from the model using the *buffer-slot-value* command in a while loop and a production in the Lisp code of the ACT-R model using a goal slot *pepper_out* for sending such a signal to the client application.

Client application with *buffer-slot-value* command:

```
while (true) {
   {method:evaluate, params:[buffer-slot-value,
   nil, goal, pepper_out], id:10}
}
```

ACT-R production with *pepper_out* goal slot:

```
(p checking-intention
   =goal>
      isa goal
==>
   =goal>
      pepper_out pepper-checks-intention
)
```

To transmit information from the robot application to the ACT-R model, the client uses the *overwrite-buffer-chunk* command to copy a chunk into the goal buffer. The model has predefined goal chunks in its declarative memory. If a predefined chunk matches the chunk from the client, all information from this model chunk is placed in the buffer and can be used to trigger a production in the model. Figure 4 illustrates the exchange of information between the robot and ACT-R.

4. Combining emotion recognition and ACT-R

Pepper has the ability to interpret the basic emotion of the human in front of the robot via facial recognition using the *ExcitementState* and *PleasureState* characteristics [18]. The *ExcitementState* can have the values *calm* or *exited*, the *PleasureState* the values *positive*, *neutral* or *negative*. Based on the work of psychologist James Russel [19], whose work focuses on emotions, a transformation matrix shown in Table 1 is used for the conversion of these states into the basic emotions *neutral*, *content*, *joyful*, *sad* and *angry*. These basic emotions should provide a sufficient basis for adapting the robot's behavior and statements to the emotional state of the human conversation partner.

The idea is to pass these findings on to an ACT-R model, which in turn draws conclusions within the framework of the human-like cognitive architecture and controls the robot application via feedback. The ACT-R model therefore controls the verbal reaction of the robot and/or an animation in the interaction with the human and adapts it to the emotion that has just been recognized. A combination of the possibilities of ACT-R with a humanoid social robot interacting directly with humans could be a way to improve the dialog between a human and a robot and make the robot appear more compassionate and empathetic.

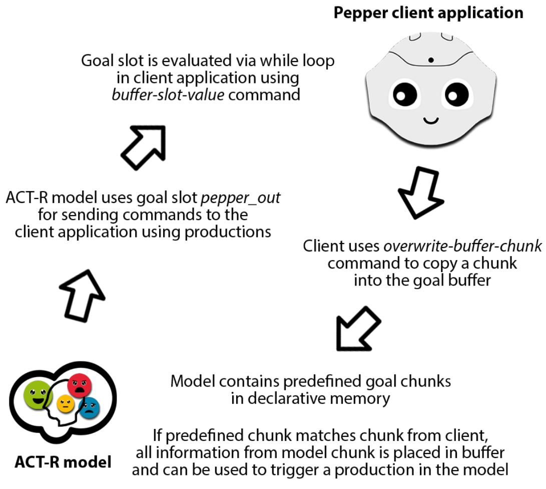
Pepper client application
Goal slot is evaluated via while loop in client application using buffer-slot-value command
ACT-R model uses goal slot pepper_out for sending commands to the client application using productions
Client uses overwrite-buffer-chunk command to copy a chunk into the goal buffer
Model contains predefined goal chunks in declarative memory
If predefined chunk matches chunk from client, all information from model chunk is placed in buffer and can be used to trigger a production in the model
ACT-R model

Figure 4: Information exchange between robot and ACT-R

Table 1
Transformation matrix to get the basic emotions

	PleasureState		
ExitementState	Positive	Neutral	Negative
Calm	Content	Neutral	Sad
Exited	Joyful	Neutral	Angry

A socket connection via the WLAN network from a robot application as a client to the dispatcher of the ACT-R application running on a PC or laptop as described in Section 3.4 enables an ACT-R model to receive and process the basic emotion values shown in Table 1 transmitted by the robot's emotion recognition. Feedback from the model to Pepper controls the robot's further behavior and the dialog. Figure 5 depicts the emotion recognition and processing by the robot and ACT-R.

For transmitting a recognized emotion the *overwrite-buffer-chunk* command is used to trigger the right productions of the ACT-R model. How the model handles the information about the person's current emotion depends on the structure of the ACT-R model with its productions and the respective application. Predefined *goal chunks* in the *declarative memory* of the model enable productions to be fired depending on the emotion values transmitted. Examples of such goal chunks, which are prepared in the Lisp code of the ACT-R model, and an example production that fills a *pepper_out* goal slot with a value that is evaluated in the client application of the robot, can be found in the following lines:

```
(add-dm
   (mood-content-chunk isa goal mood content state
       pepper-changes-mood)
```

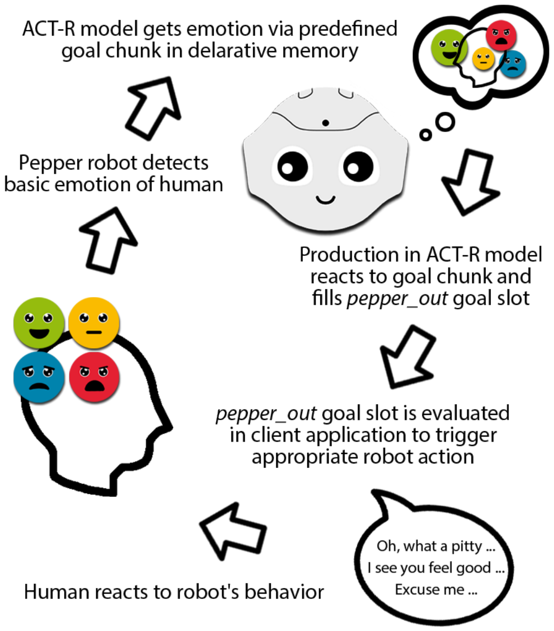
ACT-R model gets emotion via predefined goal chunk in delarative memory
Production in ACT-R model reacts to goal chunk and fills *pepper_out* goal slot
pepper_out goal slot is evaluated in client application to trigger appropriate robot action
Oh, what a pitty ...
I see you feel good ...
Excuse me ...
Human reacts to robot's behavior
Pepper robot detects basic emotion of human

Figure 5: Emotion recognition with Pepper and ACT-R

```
   (mood-joyful-chunk isa goal mood joyful state
        pepper-changes-mood)
   (mood-sad-chunk isa goal mood sad state pepper-
        changes-mood)
   (mood-angry-chunk isa goal mood angry state
        pepper-changes-mood)
)

(p pepper-content
   =goal>
      isa       goal
      mood      content
      state     pepper-changes-mood
==>
   =goal>
      pepper_out  pepper-content
      state       pepper-changed-mood
)
```

The robot's statements, which are controlled via the *Chat feature* of the client application and saved in dialog topic files as explained in Section 3.3.1, can be influenced in this way. Depending on the goal slot value, different dialogs, responses and/or animations can be triggered. The while loop that runs continuously in the client application essentially contains the following functionalities and simple IF queries for assigning the basic emotions from Pepper's emotion recognition to model chunks, evaluating the goal slot *pepper_out* of the ACT-R model and selecting the corresponding text passage in the topic file:

```
while (true) {
   // Setting chunks by Pepper's ExcitementState
        and PleasureState characteristics
   if ((MainActivity.humanPleasure == "POSITIVE")
        && (MainActivity.humanExcitement == "CALM")
        ) {
      pepperMoodAction = "mood-content-chunk"
   } else if ((MainActivity.humanPleasure == "
        POSITIVE") && (MainActivity.humanExcitement
         == "EXCITED")) {
      pepperMoodAction = "mood-joyful-chunk"
   } else if ((MainActivity.humanPleasure == "
        NEGATIVE") && (MainActivity.humanExcitement
         == "CALM")) {
      pepperMoodAction = "mood-sad-chunk"
   } else if ((MainActivity.humanPleasure == "
        NEGATIVE") && (MainActivity.humanExcitement
         == "EXCITED")) {
      pepperMoodAction = "mood-angry-chunk"
   }
   ...
   // Copy a chunk into the goal buffer and trigger
         the right productions of the ACT-R model
        using overwrite-buffer-chunk command
   {method:evaluate, params: [overwrite-buffer-
        chunk, nil, goal, pepperMoodAction], id:50}
   ...
   // Permanent evaluation of goal slot pepper_out
   {method:evaluate, params: [buffer-slot-value,
        nil, goal, pepper_out], id:10}
   ...
   // A variable bufferSlotValueOut contains the
        current value of the goal slot pepper_out
        transmitted by the ACT-R model and sets a
        corresponding variable in the client
        application
   if (bufferSlotValueOut == "PEPPER-CONTENT") {
      MainActivity.modelMood = "CONTENT"
   } else if (bufferSlotValueOut == "PEPPER-JOYFUL
        ") {
      MainActivity.modelMood = "JOYFUL"
   } else if (bufferSlotValueOut == "PEPPER-SAD") {
      MainActivity.modelMood = "SAD"
   } else if (bufferSlotValueOut == "PEPPER-ANGRY")
         {
      MainActivity.modelMood = "ANGRY"
   }
   ...
   // React to the model and go to a bookmark
        section in topic file to speak the
        appropriate text
   if (MainActivity.modelMood == "CONTENT") {
      qiChatbot.async()?.goToBookmark(topic.
           bookmarks["intention_content"]
   } else if (MainActivity.modelMood == "JOYFUL") {
      qiChatbot.async()?.goToBookmark(topic.
           bookmarks["intention_joyful"]
   } else if (MainActivity.modelMood == "SAD") {
      qiChatbot.async()?.goToBookmark(topic.
           bookmarks["intention_sad"]
   } else if (MainActivity.modelMood == "ANGRY") {
      qiChatbot.async()?.goToBookmark(topic.
           bookmarks["intention_angry"]
   }
}
```

5. Conclusion

Our proof-of-concept application shows that a coupling of ACT-R and a social robot is possible and relatively easy to implement and that the transmission of emotion data and their evaluation by an ACT-R model as well as a control of the robot via the ACT-R model works. This was achieved by directly connecting the robot application to ACT-R without using additional frameworks.

The fact that the robot can be controlled via a cognitive architecture opens up a wide range of possibilities that these architectures offer in terms of better situated human perception and improved adaptability to the behavior of a human conversation partner. However, it remains important to consider whether the effort required for implementation, modeling and resilience is appropriate in relation to the achievable functionality.

6. Prospects and further ideas

The use of a cognitive architecture in conjunction with a social robot offers far-reaching possibilities for the joint creation of added value in terms of robot behavior that is as easy as possible for humans to understand and comprehend. A dynamic person model, which reacts flexibly and as accurately as possible to changes in the behavior of a human interaction partner and adapts based on human-like cognitive rules and experiences, enables interaction experiences on common ground between humans and robots.

Enriching the cognitive model with real world data, which the robot perceives via its sensors, in turn enables the model to react to the outside world. The robot's body serves as the executive organ of the cognitive model. Ultimately, the overall result can only be as good as the quality of perception by the sensors and the possibilities offered by the robot. The Pepper robot's emotion recognition via facial expression and voice tones is always a snapshot and not perfectly reliable. Sometimes it is simply wrong or misinterprets a brief irritation on the part of the human. Therefore, ways and means must be devised for the cognitive person model to deal with these possibly contradictory impressions and draw appropriate conclusions from them.

Our first test study on this in the assumed scenario of an public authority with changing courses has shown that the participants perceive changes in the robot's behavior from case to case depending on the course and the emotional reactions of the participant. The next steps would be to develop a more extensive scenario and a more sophisticated ACT-R model in order to conduct more detailed studies.

Another promising idea might be the use of large language models (LLMs) such as ChatGPT with their ability to generate human-sounding answers to almost any question for interaction and collaboration between humans and machines. Prompt generation is the key to successful use. It is conceivable to generate prompts for LLMs with the help of a cognitive architecture from an ACT-R model. This would combine human-like cognition with human-like language skills and could – in combination with emotion recognition – perhaps evoke something like empathetic reactions from the robot and make an interaction on the path to real understanding even more pleasant for the human.

References

[1] A. Werk, S. Scholz, T. Sievers, N. Russwinkel, How to provide a dynamic cognitive person model of a human collaboration partner to a pepper robot, Society for Mathematical Psychology, 2024 forthcoming.

[2] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, Y. Qin, An integrated theory of the mind 111 (2004) 1036–1060. doi:10.1037/0033-295X.111.4.1036.

[3] F. E. Ritter, F. Tehranchi, J. D. Oury, Act-r: A cognitive architecture for modeling cognition 10 (2019). doi:10.1002/wcs.1488.

[4] F. Förster, M. Romeo, P. Holthaus, L. Wood, C. Dondrup, J. Fischer, F. Liza, S. Kaszuba, J. Hough, B. Nesset, D. Hernandez Garcia, D. Kontogiorgos, J. Williams, Working with troubles and failures in conversation between humans and robots: workshop report, Frontiers in Robotics and AI 10 (2023). doi:10.3389/frobt.2023.1202306.

[5] A. Weidemann, N. Russwinkel, The role of frustration in human–robot interaction – what is needed for a successful collaboration?, Frontiers in Psychology 12 (2021). doi:10.3389/fpsyg.2021.640186.

[6] A. Bono, A. Augello, G. Pilato, F. Vella, S. Gaglio, An act-r based humanoid social robot to manage storytelling activities, Robotics 9 (2020) 25. URL: http://dx.doi.org/10.3390/robotics9020025. doi:10.3390/robotics9020025.

[7] A. Augello, G. Pilato, An annotated corpus of stories and gestures for a robotic storyteller, 2019, pp. 630–635. doi:10.1109/IRC.2019.00127.

[8] A. Augello, G. Città, M. Gentile, A. Lieto, A storytelling robot managing persuasive and ethical stances via act-r: An exploratory study, International Journal of Social Robotics (2021). doi:10.1007/s12369-021-00847-w.

[9] G. Trafton, L. Hiatt, A. Harrison, F. Tanborello, S. Khemlani, A. Schultz, Act-r/e: An embodied cognitive architecture for human-robot interaction, Journal of Human-Robot Interaction 2 (2013) 30–55. doi:10.5898/JHRI.2.1.Trafton.

[10] S. Xu, D. Tu, Y. He, S. Tan, M. Fang, Act-r-typed human–robot collaboration mechanism for elderly and disabled assistance, Robotica 32 (2014) 711–721. doi:10.1017/S0263574713001094.

[11] A. Pipitone, A. Chella, What robots want? hearing the inner voice of a robot, iScience 24 (2021) 102371. doi:10.1016/j.isci.2021.102371.

[12] A. Pipitone, A. Chella, Robot passes the mirror test by inner speech, Robotics and Autonomous Systems 144 (2021) 103838. doi:10.1016/j.robot.2021.103838.

[13] Raluca Budiu, ACT-R / About, Technical Report, 2024. URL: http://act-r.psy.cmu.edu/about/.

[14] Aldebaran, United Robotics Group and Softbank Robotics, Pepper, Technical Report, 2024. URL: https://www.aldebaran.com/en/pepper.

[15] J. Fink, Anthropomorphism and Human Likeness in the Design of Robots and Human-Robot Interaction, Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-34103-8_20.

[16] Aldebaran, United Robotics Group and Softbank Robotics, Pepper SDK for Android, Technical Report, 2024. URL: https://qisdk.softbankrobotics.com/sdk/doc/pepper-sdk/index.html.

[17] QiSDK, Chat, Technical Report, 2024. URL: https://qisdk.softbankrobotics.com/sdk/doc/pepper-sdk/ch4_api/conversation/reference/chat.html.

[18] QiSDK, Mastering Emotion detection, Technical Report, 2024. URL: https://qisdk.softbankrobotics.com/sdk/doc/pepper-sdk/ch4_api/perception/tuto/basic_emotion_tutorial.html.

[19] J. A. Russell, Emotion, core affect, and psychological construction, Cognition and Emotion 23 (2009) 1259–1283. doi:10.1080/02699930902809375.