

# Enriching Robot's Actions with Affective Movements

Julian M. Angel-Fernandez<sup>1</sup> and Andrea Bonarini<sup>2</sup>

**Abstract**—Emotions are considered by many researchers as beneficial elements in social robotics, since they can enrich human-robot interaction with non-verbal clues. Although there have been works that have studied emotion expression in robots, mechanism to generate emotion are highly integrated with the rest of the system. This limits the possibility to use these approaches in other applications. This paper presents a system that has been initially created to facilitate the study of emotion projection, but it has been designed to enable its adaptation in other fields. The emotional enrichment system has been envisioned to be used with any action decision system. A description of the system components and their characteristics are provided. The system has been adapted to two different platforms with different degrees of freedom: Keepon and Triskarino.

## I. INTRODUCTION

The development of fast, cheap, and reliable electronics has enabled the creation of new devices and versatile robotic platforms. These new platforms' capabilities have expanded the frontiers of the robots applications to new environments where robot are expected to interact with humans, such as health care, and house cleaning, among others. However, bringing robots in these environment raises the challenge to increase robots' acceptance. Although this could be seen as an easy task that just would need improvements in robots' appearances and capabilities, it is possible that people would expect to treat robots as humans as they do with computers. [1], which makes necessary the creation of robots able to fulfil this expectations.

Some researchers have suggested that embedding emotion expression capabilities to robots could improve their acceptance in social environments [2]. To this purpose, some researchers [3], [4] have added specific emotional poses and expression to their robots. Others have studied how to convey emotions with specific platforms [5], [6]. Nevertheless, these works have created modules to show emotions that are strongly integrated to their solutions, which eliminate the possibility to re-use or adapt their systems into other projects.

In theory, the projection of emotion with humanoid embodiments could be simplified by mimicing the human movements. However, this idea cannot be fully implemented, due physical limitations in robots. Since an exact matching of human movements to convey emotions could not be used in robots [7], [8], some researchers are studying diverse features and values to express emotions with different platforms. As

a consequence, these results could not be widely used due to the difference among the platforms.

This paper presents an Emotional Enrichment System (EES) that modifies actions' parameters and adds additional actions to create the illusion of emotion expression in a robot. Although the EES was originally conceived to be used to facilitate the study of emotion projection, its design was devised to make it extendable to other platforms and adaptable to new tasks. To achieve this goal, the system relies on an Emotional Execution Tree (EXT), which is based on simple actions, sequential and parallel nodes. Additionally, the concept of compound actions is used to group a bunch of nodes, thus reducing the tree dimension and allowing the reuse of recurrent actions generated by specific combinations of simple actions and other nodes. This EXT has been formalized to give a guideline to further implementations and extensions.

The rest of the paper is organized as follows. Section II provides a brief overview of relevant work. Section III introduces all the concepts that are used in the system. Section IV gives the basic formalization of our system and principal components terms used on it. Section V presents the EBFN created to specify the files used by the system. Section VI describes the implementation of the system and shows two demonstrations done with the system using platforms with different capabilities.

## II. RELATED WORK

The use of emotion enrichment to improve human robot interaction is not new. There have been several researchers that have enhanced their social robots with emotions or studied how to convey emotions in robotic platforms [5], [6]. One of the first, well-known expressive robots is Kismet [3], a robotic face able to interact with people and to show emotions. This platform uses a specific set of movements based on the Ekman's studies on human emotion expression [9]. Other approaches have tried to use anthropomorphic [4] and human-like platforms to convey emotions to study the response of people towards the robot. However, the emotions portrayed were hand-coded, hard-wired to the respective platforms, and their parametrization is not available.

On the other hand, studies focused on entertainment robotics have tried to introduce emotional actions to improve the audience's experience. Breazeal and collaborators [10] used one robot on the stage. This anemone-like robot had few behaviours, which included getting scared when a person comes too close; the robot was able to show some basic emotions (i.e., fear and interest). Knight [11] used the platform NAO to produce a sort of stand-up comedy. The robot

<sup>1</sup>Julian M. Angel-Fernandez is Researcher at Automation and Control Institute, Vienna University of Technology, Vienna, Austria  
julian.angel.fernandez@tuwien.ac.at

<sup>2</sup>Andrea Bonarini is Full Professor at the Department of Electronics, Information, and Bioengineering, Politecnico di Milano, Milan, Italy,  
andrea.bonarini@polimi.it

performs basic actions to add some expressiveness to the joke, but it is not intended to project any emotion. Trying to add some theatrical realism, Breazeal and collaborators [12] designed and implemented a system to control a lamp. The main characteristic of this lamp is that it could be controlled by just one person, by selecting pre-coded emotions.

Other works in performance robotics have developed systems that do not convey any kind of emotion as *Roboscopia* [13], Fan and collaborators [14], and adaptation of Shakespeare's *Midsummer Night's Dream* [15] use robots in performances with real actors, but without any emotional expression. Although these works have expressed emotions in their studies, they were not focus on create a framework that could be used by others researchers in HRI. Therefore, the use of affective architectures for virtual agents (e.g. SAIBA/BML framework [16]) could be though as a feasible solution. However these framework describe high level actions without given relevant details to its implementation in physical platform. This gap cannot be closed by making a direct map the abstract actions and the physical platforms. This is due the fact that virtual agents do not have physical constraints, while robots are constrained by their physical capabilities [7], [8].

### III. BASIC CONCEPTS

The system is based on six main concepts that structures and enables the enrichment of actions with emotions. (i) *Action message* establishes the structure of the message to describe any kind of action (i.e., simple and compound). This message also specifies how the actions are executed (i.e., in parallel or in sequence) and which action is predominant (i.e., primary or secondary). (ii) *Emotional parameters* describe how the emotional enrichment should be done to convey a specific emotion in a specific simple action. This description could also include addition of other simple actions and vary over time. (iii) *Character description* enables the possibility to establish how to modify emotional expressions to generate diverse "rhythms". (iv) *Emotional Execution Tree* (EXT) is a computational representation of desired actions that should be executed. This tree is created from the action message description and then modified using the emotional parameters and character description. The remaining two concepts are explained below.

#### A. Emotion Parameters

The emotion parameters specify how actions parameters should be modified in order to express an emotion. However, there is going to be a diverse set of parameters depending on the nature of the action. For example, consider the action move body and speak. The human action of speaking is related to changes in the vocal cords to produce different sounds, while move body is directly related to the body movement. The speaking action would have the following parameters: text, pitch and tone. On the other hand, the action move body would have as parameters the desired destination, velocity, and trajectory constraints. Since each action is characterized by different parameters, their modification must be

TABLE I: Description of the seven simple actions implemented, and their respective parameters. Where P is 2D position, V is 2D velocity vector and angular velocity, and T is time

Action Name	Description	Parameter(s)
Do nothing	It waits for a time $t$ before it is terminated. It could be seen as a delay.	$T$
Move body	It moves the platform from its current position $a$ to a desired position $b$ .	$P$ , and $V$
Oscillate body	It generates an oscillation in the whole platform by an angle $\theta$ .	$\theta$ and $V$
Move shoulder	It moves the shoulders to a desired angle $\theta$ . It is considered as angular movement.	$\theta$ and $V$
Oscillate shoulder	It oscillates the shoulders by a given angle $\theta$	$\theta$ and $V$
Move torso	It moves the torso to a desired angle in <i>yaw</i> , <i>pitch</i> and <i>roll</i>	<i>yaw</i> , <i>pitch</i> , <i>roll</i> and $V$
Oscillate torso	It oscillates the shoulders by a given angle $\theta$	$\theta$ and $V$

done accordingly to the parameters. Therefore, a variety of emotional parameters would be necessary to describe the modification of all type of actions. So, it was decided to focus on the parameters that could be used to express emotions with the Triskarino and Keepon. It was determined that these parameters could just use movement actions, which are described by position, velocity and acceleration.

#### B. Simple and Compound Actions

In order to generate a system that could be used in diverse platforms, an abstraction level in which all of them could fall is required. Therefore simple and compound actions are used to achieve this goal. Simple actions are actions that are considered as primitives: they are used as building blocks. As a consequence, these actions are described in the system and are the ones in which the parameters are changed to embed the emotions. Their description specifies mandatory and optional parameters that are required to execute an action. These parameters are modified accordingly to the information provided in the emotional parameters. Compound actions are actions that are created from simple actions and other compound actions. These actions are not implemented in the system, but, if it is needed, they can be described in it (e.g., compound actions that are used often).

A set of simple actions to be implemented were selected to test the system. These actions were selected considering the capabilities of Keepon and Triskarino. The eight actions selected were: move body, oscillate body, move shoulder, oscillate shoulder, move torso, oscillate torso, and do nothing. Description for each action, its mandatory parameters and optional parameters are presented in Table I.

To exemplify the use of these actions, let's consider a sequence of movements illustrated in Figure 1. This sequence of movements correspond to an adaptation of the first part of the balcony scene of Shakespeare's *Romeo and Juliet* play [17]. The first part of this scene is characterized by

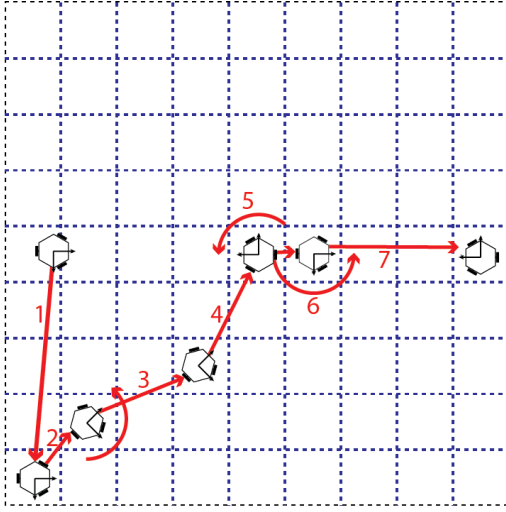


Fig. 1: Sketch of Romeo's movements for the balcony scene in Romeo and Juliet play. The arrows show the direction of the movements, and the numbers their sequence.

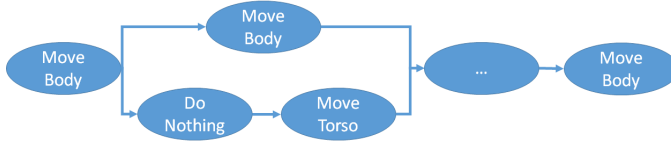


Fig. 2: Sequence of simple actions for the movements depicted in Figure 1.

strong changes in Romeo's emotional state, which is the perfect test for the system. However, just body movements would not exploit the other movements of the platform neither test the whole system. Thus, additional movements are added to illustrate and test the system. A partial view of the whole sequence of actions is illustrated in Figure 2. The first node correspond to the first action in Figure 1. The second actions corresponds to a compound action created by the parallel execution of the simple action move body and other compound action.

#### IV. EMOTIONAL TREE

Emotional Execution Tree is a connected acyclic graph  $G(V, E)$  with  $|V|$  vertexes and  $|E|$  edges. The root and non-leaf nodes could be of either *parallel* or *sequential* type, which represent the type of actions that would be executed by the system. Sequential nodes execute one branch after the other. Once all the branches have been executed, it notifies its predecessor to continue with the execution of other branches. The parallel node executes all the branches at the same time. This type of node offers two levels of synchronization: action and emotion. The action level synchronization enables the possibility to stop all the branches once the main branch has completed its execution. This main branch is established in

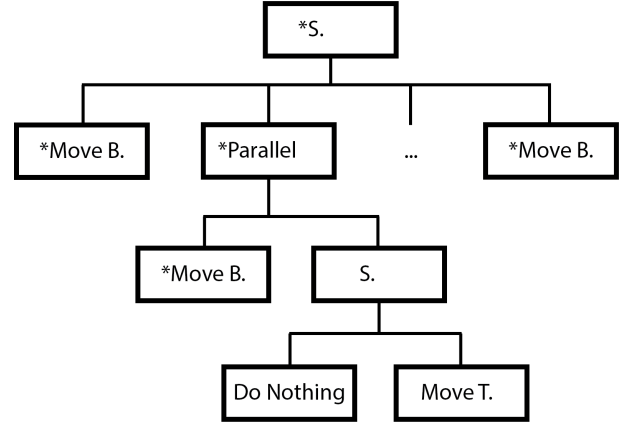


Fig. 3: Emotional Execution Tree for the sequence of actions depicted in Figure 2. The \* symbolize that node is principal. S. represents sequential.

the action message. On the other hand, the emotion level synchronization update to a new emotional parameter of all the branches once the main branch updates to a new emotional parameter. In both cases, any update or finish message will be ignored if the branch is not principal. These two levels of synchronization creates four sub-types of parallel nodes: (i) action and emotion synchronous, (ii) action synchronous and emotion asynchronous, (iii) action asynchronous and emotion synchronous, or (iv) action and emotion asynchronous.

Finally, leaf nodes are only simple action nodes that have been implemented in the system. Any node can belong to one of two levels: principal or secondary. If a node is principal, it will notify its predecessor about the messages that it has received, while the secondary node cannot propagate any message to its predecessor. This enables the possibility to have actions that have priority over others. To exemplify this, lets continue with the sequence of actions depicted in Figure 2. The Emotional Execution Tree for that sequence of actions would be like the one presented in Figure 3.

#### A. Formalization of Simple and Compound Actions

Formalizing simple and compound actions generates a common system to design them and verify if they are described correctly. Simple actions ( $SA$ ) could be seen as functions that map a set of parameters  $p_i^{pl} \in P$  to specific platform ( $pl$ ) movements ( $m_j^{pl} \in M^{pl}$ ) in the set of movements for that platform ( $m_j^{pl} = sa^{pl}(\{p_i^{pl}\})$ ). Each  $sa^{pl}$  could have different implementations due to different reasons such as: platform's type, action's goal, how the mapping from inputs to outputs are done, among others. In our case, we consider that two actions are different if  $(\forall sa_i, sa_j \in SA, \forall q, t \in PL | Param(sa_i^q) \neq Param(sa_j^t) \Rightarrow sa_i^q \neq sa_j^t)$ , where  $PL$  is the set of all the possible platforms,  $Param(\cdot)$  returns the set of parameters that are input of the  $sa$  (e.g.,  $vel, pitch, text$ ). On the other hand, we will consider two actions as equivalents if  $(\forall sa_i, sa_j \in SA, \forall q, t \in PL | Param(sa_i^q) = Param(sa_j^t) \wedge Obj(sa_i^q) =$

$Obj(sa_j^t) \Rightarrow sa_i^q \equiv sa_j^t$ ), where  $Obj(\cdot)$  returns the set of objectives that are intended to be achieved by a given action. This  $SA$  equivalence allows us to avoid the specification of all the platforms for all the actions that share the same objective and hide leaf nodes that do not give relevant information to understand the emotional tree.

With this  $SA$  definition, it is now possible to define emotion enrichment as  $(\forall sa \in SA, \forall e \in E, \exists en \mid en = Enrichment(sa, e, Param(sa), Intensity(e)))$ , where  $Intensity(\cdot)$  returns the intensity of a given emotion,  $E$  is the set of emotions, and  $en$  can be the same  $sa$  but with the parameters modified in order to convey the desire emotion  $e$ , or even a set of  $sa$  with their parameters modified. This definition goes along with the idea that new  $sa$  could be added to convey a specific emotion. In addition, the  $Enrichment(\cdot)$  function has the following properties:

- $(\forall sa_i, sa_j \in SA, \forall e \in E \mid sa_i \equiv sa_j \Rightarrow Enrichment(sa_i, e, \dots) \equiv Enrichment(sa_j, e, \dots))$
- $(\forall sa_i, sa_j \in SA, \forall e \in E \mid (sa_i \not\equiv sa_j) \Rightarrow Enrichment(sa_i, e, \dots) \not\equiv Enrichment(sa_j, e, \dots))$

The concept of compound action ( $CA$ ) is defined as  $g = ca(\{p_i\})$ , where  $(p_i \in Param(sa))$ ,  $(sa \subseteq SA)$ ,  $(g \in EXT)$ .

## V. DESCRIBING COMPONENTS

The description files allow the parametrization and extent the system to fit other applications is required. To standardized the mechanisms to describe these files, it was decided to create a language to describe them. This allows other researchers to create their own files and extended the language to fit their needs. It was decided to use EBNF to describe language.

### A. Emotion Description

The language created to describe emotion descriptors is the following:

```

<emotion description> → "{" "emotion:" <string> "," "observation:" <string> "," <action> ("," <action>)* "}"
<action> → <string> ":" {" <description> "," <actions affected> "}"
<description> → "description: {" <action name> "," <emotion> "," <parameter's type> "}"
<action name> → "emotionProfileAction:" <string>
<emotion> → "emotionProfileEmotion:" <string>
<parameter's type> → "movement_parameter"
<actions affected> → "actions: {" <action parameter> ("," <action parameter>)* "}"
<action parameter> → <string> ":" {" <reference> "," <repetition> "," <parameters> "}"
<reference> → "reference:" <number>
<repetition> → "repetition:" "yes" | "no"
<parameters> → "parameters: [" <parameter description> ("," <parameter description>)* "]"
<parameter description> → <movement parameter description>
<movement parameter description> → "{" "time:" <number> "space:" <number> "}"

```

### B. Action Message

The language created to describe action Messages is the following:  $\langle action \rangle \rightarrow \langle simple action \rangle \langle compound action \rangle \langle context \rangle$

```

<simple action> → "{" <action header> "," "parameters: [" <simple action parameters> "]" }"
<compound action> → "{" <action header> "," "parameters: [" <compound action parameters> "]" }"
<action header> → "type:" <action type> "," "name:" <string> "," (is primary)
<simple action parameters> → "{" <parameter header> "," <parameter description> "}"
<compound action parameters> → <simple action parameters> ("," <simple action parameters>)*
<context> → "{" <action type> "," <emotion sync> "," <action sync> "," <is primary> "," "information:" <string> "," "actions:" <action> "}"
<parameter header> → "type:" <parameter type> "," "name:" <string>
<parameter description> → <parameter amplitude> | <parameter circle> | <parameter landmark> | <parameter point> | <parameter speech> | <parameter square> | <parameter time>
<parameter type> → "mandatory_parameter" | "optional_parameter"
(is primary) → "yes" | "no"
<emotion sync> → "yes" | "no"
<action type> → "parallel_node" | "sequential_node" | "simple_action" | "composite_action"

```

## VI. IMPLEMENTATION

The system was implemented in C++ and interfaced with ROS. The design (Fig. 4) was created following the description presented in previous sections. The emotion enrichment core is divided in three different modules. Each module is responsible for one of the following phases:

- 1) *Generation of emotional execution tree*: this phase starts every time that a new action message is received. The process begins by parsing the format, verifying that the actions described on it exist in the system, and that the parameters correspond to the ones expected by each action included in the message. This parameters' verification is done on the implemented description for each action, which describes the parameters that are mandatory and those that are optional. When the verification is done, and all the actions exist and the parameters correspond, an Emotional Execution Tree (EXT) is created such as the one presented in Figure 3
- 2) *Emotion addition*: uses the EXT created in the previous phase. In this phase new simple actions ( $sa$ ) are added to the EXT, and the  $sa$ 's parameters are modified following the emotion description, which is loaded from files. This process is broken down in two steps. First, all the actions that are required to convey the desired emotion, and that are not yet present are added. Second, the emotional parameters are modulated basing on the emotion's intensity and character traits. The

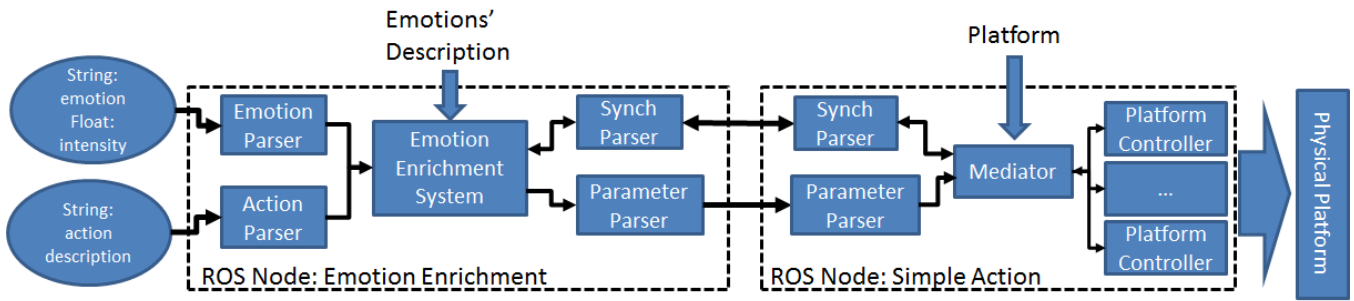


Fig. 4: General system design. Each simple action corresponds to one ROS node, and there is just one node for the emotion enrichment system. The ovals represent the ROS topic parameters, rectangles represent *black boxes*, and texts outside containers represent input files that contain the system parametrization.

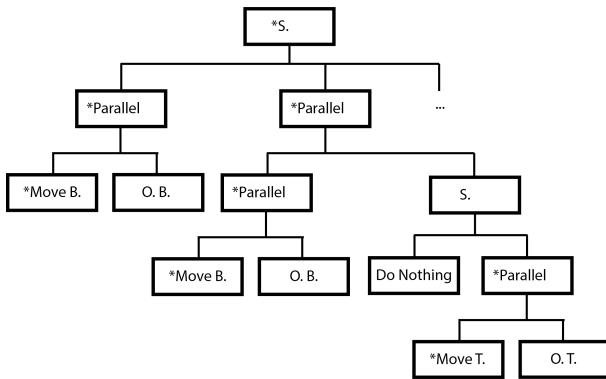


Fig. 5: Enrich Emotional Execution Tree.



Fig. 6: Keepon platform.

final EXT is presented in Figure 5.

- 3) *Execution*: this is the last phase and it is done after the *ext* is "coloured" with emotional characteristics (actions additions and emotional parameters). The decision to have two different communication channels, one for action parameters and another for the action emotional parameters, was taken to enable the possibility to update the emotional parameters without interfering with the current execution. In this phase a reference to the principal and emotional action is maintained, thus, when a new emotion is to be expressed, the system stops the previous emotional action and starts executing the new ones without affecting the general execution of the actions.

All the text message broadcast among the nodes are written using JSON format, which is human understandable, light, and there are many available parsers.

To test the system two different platforms were used: Keepon Pro and an own made platform called Triskarino. Keepon has been used to test the interoperability of the system to different platforms, while Triskarino has been used in diverse case studies to explore emotion projection with a non anthropomorphic platform.

#### A. Keepon Test

To test the system with Keepon (Fig. 6) was just necessary to implement the platform's controllers in each one of the

simple action ROS nodes. Given that this platform does not have the capability to displace its body, the action move body was not implemented. Once added these controllers to the system, we proceeded to modify the configuration files to use this platform instead of Triskarino. With this small modification, we were able to change from one platform to other. In the video<sup>1</sup> is shown the test where Keepon had to move its torso forward and backward to express a "happy" emotion. The action is given by console telling the robot to bend the torso to a desired angle in x. The torso oscillation in y is added automatically by the system following the description given to happiness.

#### B. Triskarino Test

The system has been widely used with Triskarino (Fig. 7) during our studies on projecting emotions with a non human like platform. During these studies the system was used in different experiments to move the robot on a straight line showing different emotions, each one selected from our command console. Moreover, to show the whole capabilities of the emotional enrichment system, a little scene was set up. The scene is a modification of the first part of the balcony scene of Shakespeare's Romeo and Juliet play [17]. The simplified sketch of our version could be seen in the Fig. 1. As it can be seen the stage was divided in 81 squares

<sup>1</sup>YouTube video name Emotional Enrichment System, url: <https://youtu.be/bRSXQ0rzK08>

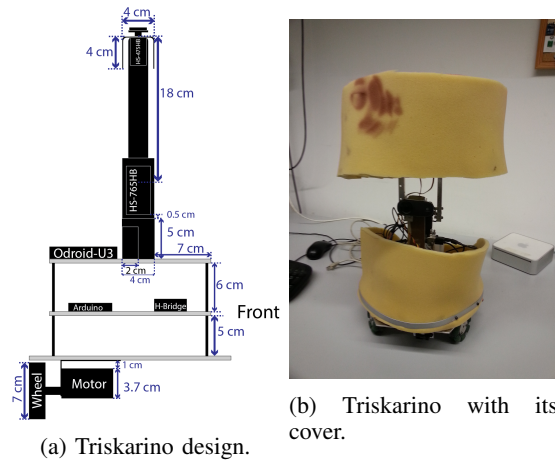


Fig. 7: Triskarino platform.

and the positions were given to the system in terms of the desire square to be reached, which allows the robot to adjust to the stage's dimension. The whole sequence of actions were specified as unique action, having several sequential context and just move body action. The other actions, such as oscillate body or blend upper body were added online accordingly the desired emotion.

## VII. CONCLUSIONS AND FURTHER WORK

An Emotional Enrichment System has been designed and implemented to enrich robots' movements with emotions. To achieve this, it was used an Emotional Execution Tree created from three different types of nodes: simple actions, parallel, and sequential. Simple actions are functions that map a set of parameters to specific movements. Sequential nodes execute the sequence of actions associated to this type of node in a sequential order, while parallel executes them all at the same time. To enable synchronization among simple actions, parallel nodes could be one of four different subtypes, and sequential just one of two different subtypes. A formalization of the Emotional Execution Tree and the principal considerations during the implementation of the system have also been provided. To show the system's versatility, it was used with quite different platforms such as a Keepon Pro and Triskarino. Keepon was used to perform simple actions, while Triskarino was used to test complex actions with different parametrizations of emotions.

The obtained results show that the system could enrich the robotic actions with emotions, which could be parametrized from configuration files. Additionally, the design of the system makes it possible to adopt it for different platforms using the same action description.

## REFERENCES

- [1] B. Reeves and C. Nass, *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places*. New York, NY, USA: Cambridge University Press, 1996.
- [2] A. Paiva, I. Leite, and T. Ribeiro, "emotion modeling for social robots", [Online]. Available: "http://www.oxfordhandbooks.com/10.1093/oxfordhb/9780199942237.001.0001/oxfordhb-9780199942237-e-029"

- [3] C. Breazeal, *Designing Sociable Robots*. Cambridge, MA, USA: MIT Press, 2002.
- [4] S. Embgen, M. Luber, C. Becker-Asano, M. Ragni, V. Evers, and K. O. Arras, "Robot-specific social cues in emotional body language," in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN'12)*. USA: IEEE Computer Society, September 2012, pp. 1019–1025.
- [5] J. Li and M. H. Chignell, "Communication of emotion in social robots through simple head and arm movements," *I. J. Social Robotics*, vol. 3, no. 2, pp. 125–142, 2011.
- [6] L. Brown and A. M. Howard, "Gestural behavioral implementation on a humanoid robotic platform for effective social interaction," in *Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, Aug 2014, pp. 471–476.
- [7] M. Saerbeck and A. J. N. van Breemen, "Design guidelines and tools for creating believable motion for personal robots," in *RO-MAN*, 2007, pp. 386–391.
- [8] A. Beck, A. Hiolle, A. Mazel, and L. Cañamero, "Interpretation of emotional body language displayed by robots," in *3rd ACM Workshop on Affective Interaction in Natural Environments*, 2010, pp. 37–42.
- [9] P. Ekman, *Emotions Revealed : Recognizing Faces and Feelings to Improve Communication and Emotional Life*. Owl Books, Mar. 2004.
- [10] C. Breazeal, A. Brooks, J. Gray, M. Hancher, C. Kidd, J. McBean, D. Stiehl, and J. Strickon, "Interactive robot theatre," in *Proceedings of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems(IROS 2003)*, vol. 4, 2003, pp. 3648–3655.
- [11] H. Knight, S. Satkin, V. Ramakrishna, and S. Divvala, "A savvy robot standup comic: Online learning through audience tracking," *TEI 2011*, January 2011.
- [12] G. Hoffman, R. Kubat, and C. Breazeal, "A hybrid control system for puppeteering a live robotic stage actor," in *RO-MAN 2008*, M. Buss and K. Kühnlenz, Eds. IEEE, 2008, pp. 354–359.
- [13] LAAS-CNRS, "Roboscopia, the robot takes the stage!" <http://www.openrobots.org/wiki/roboscopia>.
- [14] C.-Y. Lin, C.-K. Tseng, W.-C. Teng, W. chen Lee, C.-H. Kuo, H.-Y. Gu, K.-L. Chung, and C.-S. Fahn, "The realization of robot theater: Humanoid robots and theatric performance," in *International Conference on Advanced Robotics, 2009. ICAR 2009.*, 2009.
- [15] R. Murphy, D. Shell, A. Guerin, B. Duncan, B. Fine, K. Pratt, and T. Zourntos, "A midsummer night's dream (with flying robots)," *Autonomous Robots*, vol. 30, no. 2, pp. 143–156, 2011. [Online]. Available: <http://dx.doi.org/10.1007/s10514-010-9210-3>
- [16] S. Kopp, B. Krenn, S. Marsella, A. N. Marshall, C. Pelachaud, H. Pirker, K. R. Thórisson, and H. Vilhjálmsón, *Towards a Common Framework for Multimodal Generation: The Behavior Markup Language*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 205–217.
- [17] R. S. Company, "Royal shakespere company - romeo & juliet, on stage footage - ny," <http://www.youtube.com/watch?v=FHoapLO6Zd8>.