Annex I: Specification of the RAPP Core Agent

Warsaw University of Technology, Institute of Control and Computation Engineering, Warsaw, Poland

1 Introduction

The proposed design method requires from the designer the specification of a specific model of a robot system executing the task that it is meant for. This model is produced on the basis of a universal model of a robotic system described below. In this approach robots in single- or multi-robot systems are represented as embodied agents. As embodied agents are the most general forms of agents, out of them any robot system can be designed. The thus produced specification is used as a blueprint for the implementation of the system. The described methodology of creating robot systems has been described in many papers, e.g. [1–6].

2 An embodied agent

A robotic system is represented as a set of agents a_j, $j = 1, \ldots, n_a$, where n_a is the number of agents (j designates a particular agent). Embodied agents have physical bodies interacting with the environment. This work focuses on embodied agents [2], but all other agents can be treated as special cases with no body, thus the presentation is general.

2.1 General inner structure of an embodied agent

An embodied agent a_j, or simply an agent, possesses real effectors E_j, which exert influence over the environment, real receptors R_j (exteroceptors), which gather information from the surroundings, and a control system C_j that governs the actions of the agent in such a way that its task will be executed. The exteroceptors of the agent a_j are numbered (or named), hence $R_{j,l}$, $l = 1, \ldots, n_R$, and so are its effectors $E_{j,h}$, $h = 1, \ldots, n_E$. Both the receptor readings and the effector commands undergo transformations into a form that is convenient from the point of view of the task, hence the virtual receptors r_j and virtual effectors e_j transform raw sensor readings and motor commands into abstract concepts required by the control subsystem to match the task formulation. Thus the control system C_j is decomposed into: virtual effectors $e_{j,n}$, $n = 1, \ldots, n_e$, virtual receptors $r_{j,k}$, $k = 1, \ldots, n_r$, and a single control subsystem c_j. The general structure of an embodied agent is presented in fig. 1. Virtual receptors perform sensor reading aggregation, consisting in either the composition of information obtained from several exteroceptors or in the extraction of the required data from one complex sensor (e.g. camera). Moreover the readings obtained from the same exteroceptors $R_{j,l}$ may be processed in different ways, so many virtual receptors $r_{j,k}$ can be formed. The control loop is closed through the environment, i.e. exteroceptor readings $R_{j,l}$ are aggregated by virtual receptors to be transmitted to the control subsystem c_j which generates appropriate commands for the virtual effectors e_j to translate into signals driving the effectors E_j. This primary loop is supplemented by links going in the opposite direction. The control subsystem c_j can both reconfigure exteroceptors R_j and influence the method how the virtual receptors r_j aggregate readings, thus a link from the control subsystem to the receptor emerges. The control subsystem also acquires proprioceptive data from the effectors. Moreover, an agent through its control subsystem is able to establish a two-way communication with other agents $a_{j'}$, $j \neq j'$.

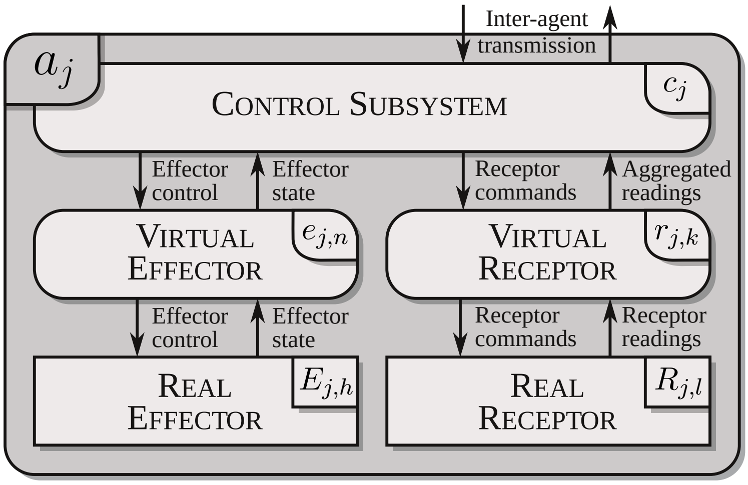

Figure 1: Structure of an embodied agent

The control subsystem as well as the virtual effectors and receptors use communication buffers to transmit or receive information to/from the other components (fig. 2). A systematic denotation method is used to designate both the components and their buffers. To make the description of such a system concise no distinction is being made between the denotation of a buffer and its state (its content) – the context is sufficient. In the assumed notation a one-letter symbol located in the centre (i.e. E, R, e, r, c) designates the subsystem. To reference its buffers or to single out the state of this component at a certain instant of time extra indices are placed around this central symbol. The left superscript designates the subsystem to which the buffer is connected. The right superscript designates the time instant at which the state is being considered. The left subscript tells us whether this is an input (x) or an output (y) buffer. When the left subscript is missing the internal memory of the subsystem is referred to. The right subscript may be complex, with its elements separated by comas. They designate the particular: agent, its subsystem and buffer element. Buffer elements can also be designated by placing their names in square brackets. For instance ${}^{e}_{x}c^{i}_{j}$ denotes the contents of the control subsystem input buffer of the agent a_j acquired from the virtual effector at instant i. Similarly functions are labeled. The central symbol for any function is f, the left superscript designates the owner of the function and the subsystem that this function produces the result of its computations for, the right superscript: τ, σ, ϵ refer to the terminal, initial and error conditions respectively (each one of them being a predicate). A missing right superscript denotes a transition function. The list of right subscripts designates a particular function. Thus the internal structure of an agent is presented in fig. 2.

2.2 General subsystem behaviour

Fig. 3 presents the general work-cycle of any subsystem s, where $s \in \{c, e, r\}$, of the agent a_j. The functioning of a subsystem s requires the processing of a transition function which uses as arguments the data contained in the input buffers ${}_{x}s_j$ and the internal memory ${}^{s}s_j$,

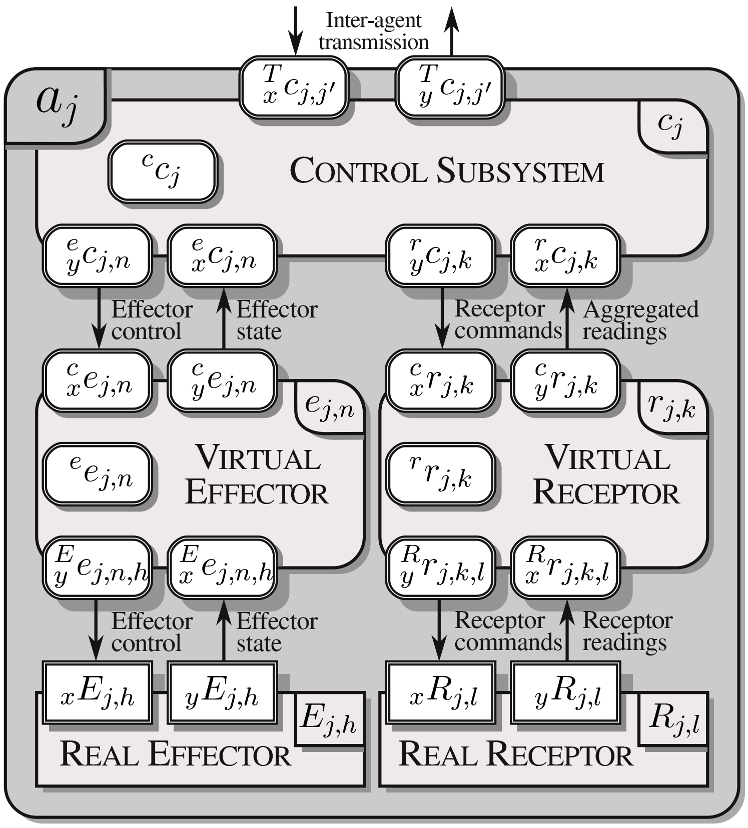

Figure 2: Internal structure of an agent a_j

to produce the output buffer values ${}_y s_j$ and new memory contents ${}^s s_j$. Hence the subsystem behaviour is described by a transition function ${}^s f_j$ defined as:

$$\left[{}^s s_j^{i+1},\ {}_y s_j^{i+1}\right] := {}^s f_j({}^s s_j^i,\ {}_x s_j^i). \tag{1}$$

where i and $i+1$ are the consecutive discrete time stamps[1] and $:=$ is the assignment operator[2]. Function (1) describes the evolution of the state of a subsystem s. A single function (1) would be too complex to define it in a monolithic form, thus it is usually decomposed into a set of partial functions:

$$\left[{}^s s_j^{i+1},\ {}_y s_j^{i+1}\right] := {}^s f_{j,u}({}^s s_j^i,\ {}_x s_j^i), \tag{2}$$

where $u = 0, \ldots, n_{f_s}$. Capabilities of the agent arise from the multiplicity and diversity of the partial functions of its subsystems. Such a prescription requires rules of switching between different partial transition functions of a subsystem, thus three additional Boolean valued functions (predicates) are required:

- ${}^s f_{j,u}^{\sigma}$ defining the initial condition,
- ${}^s f_{j,u}^{\tau}$ representing the terminal condition and
- ${}^s f_{j,u}^{\varepsilon}$ representing the error condition.

[1]It should be noted that although all the subsystems use the same symbol to represent the discrete time, i.e. i, in reality each one of them will have a different sampling period, thus they should be represented differently. However using a different symbol for each subsystem would rather produce confusion or require extra indices, thus making the outcome less intelligible. The time stamp of the initial step of execution of any behaviour of any subsystem is symbolised by i_0.

[2]The assignment operator ":=" in the specification of the Core Agent is abbreviated to just the symbol "="

The first one selects the transition function for cyclic execution, while the second determines when this cyclic execution should terminate. The last one detects that an error has occurred. Hence a multi-step evolution of the subsystem in a form of a behaviour $\mathcal{B}_{j,u}$ is defined as:

$$ {}^{s}\mathcal{B}_{j,u} \triangleq {}^{s}\mathcal{B}_{j,u}\left({}^{s}f^{\sigma}_{j,u},\ {}^{s}f^{\tau}_{j,u},\ {}^{s}f^{\varepsilon}_{j,u}\right) \tag{3} $$

The execution pattern of such a behaviour is presented in fig. 3. The $s_{j^\bullet}$, where $j^\bullet \in \{j, j'\}$, denotes all subsystems associated with s_j (in the case of the control subsystem some of those subsystems even may not belong to the same agent, hence j' appears). In reality an error condition ${}^{s}f^{\varepsilon}_{j,u}$ does not have to be specified explicitly, as during code implementation any error detected by the program will terminate the execution of the behaviour in which it occurred, thus the same result will be attained.

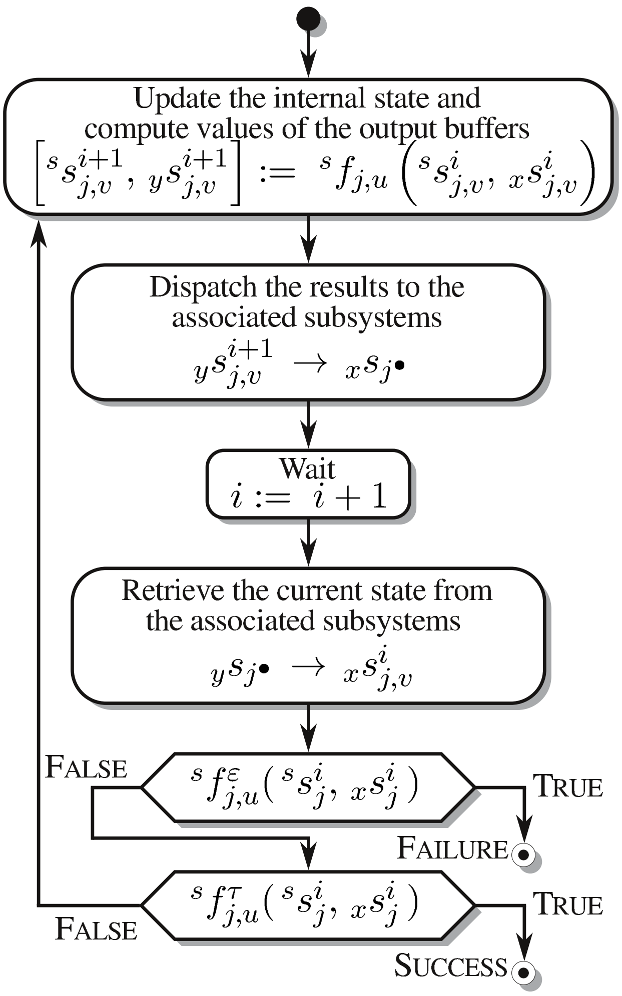

Figure 3: General flow chart of a subsystem behaviour ${}^{s}\mathcal{B}_{j,u}$, where $\bullet$ represents any subsystem including another agent

The behaviours ${}^{s}\mathcal{B}_{j,u}$ can be associated with the nodes of a graph and initial conditions with its arcs, thus a finite state automaton representation results (fig. 4). The set of initial conditions singling out the next behaviour to be executed can be used to define a state transition table of the automaton. Behaviour selection represented by a hexagonal block is executed as a stateless switch defined by the initial conditions ${}^{s}f^{\sigma}_{j,u}$. ${}^{s}\mathcal{B}_{j,0}$ is the default (idle) behaviour, activated when no other behaviour can be activated.

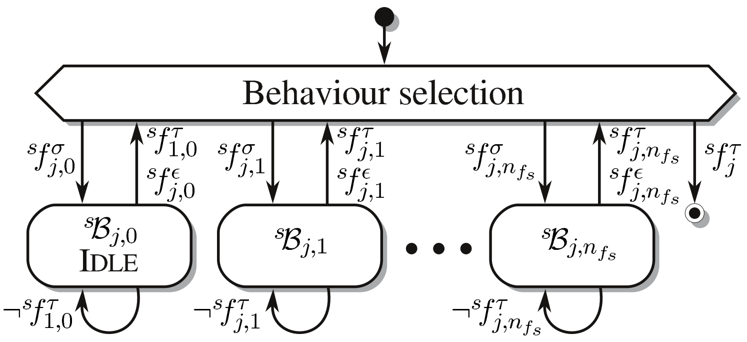

Figure 4: State graph of the behaviour selection automaton

3 Core agent a_{core}

The core agent agent a_{core} is described as a general embodied agent.

3.1 Virtual effector $e_{\text{core,body}}$

The virtual effector $e_{\text{core,body}}$ is responsible for controlling the body motions. It should be noted that all the activities performed by the NAOqi functions are executed within the real effector, thus NAOqi library is treated as an element of the real effector $E_{\text{core,body}}$. Below the contents of transmission buffers as well as the transition functions and terminal conditions of the behaviours of the virtual effector are defined. A dash in the definition of a transition function implies that a certain output variable or a certain set of output variables is not assigned a value. This further implies that the values of those variables are not sent to an associated subsystem. Usually such variables are not mentioned in the definition of the transfer function, but if they appear in some iterations of the behaviour and in some they do not, for the purpose of completeness the lack of value assignment is signalled by a dash. Similarly, if a certain behaviour is based on a transition function that produces no output values, this is signalled by a dash, otherwise one could come to a false conclusion that there exist behaviours not based on transition functions. The global frame is initialized when the robot is turned on and is fixed with the robot initial position.

3.1.1 Communication buffers and internal memory of the virtual effector $e_{\text{core,body}}$

- Internal memory ${}^{e}e_{\text{core,body}}$:
  - dja – desired joint angles with parameters (memorized argument of the INTERPOLATION command)],
    $\text{dja} = [\text{joints}, \text{values}, \text{fractionMaxSpeed}]$, where:
    - joints – a name or names of joints,
    - values – one or more angles in radians,
    - fractionMaxSpeed – fraction of the maximum speed during the angles interpolation,
  - tm – termination marker
  - pose – current robot position estimated by the Extended Kalman Filter (EKF),
    $\text{pose} = [\text{x}, \text{y}, \theta]$, where:
    - x – current robot position with respect to the X-axis of the global coordinate system, in meters,
    - y – current robot position with respect to the Y-axis of the global coordinate system, in meters,
    - θ – current angle around the Z-axis, in radians.

- Real effector control ${}^{E}_{y}e_{\text{core,body}}$:[3]

cmd – command from the virtual effector,
data_name – data name of a parameter stored in a NAOqi ALMemory module,
velocity – velocity of motion with respect to the robot coordinate frame (memorized argument of the MOVE command);
velocity = $[v_x, v_y, \omega]$, where:
  v_x – velocity along the X-axis, in meters per second,
  v_y – velocity along the Y-axis, in meters per second,
  ω – velocity the around Z-axis, in radians per second,
dpose – desired position with respect to the robot coordinate frame (memorized argument of the MOVE command);
dpose = $[x, y, \theta]$, where:
  x – distance along the X-axis, in meters,
  y – distance along the Y-axis, in meters,
  θ – rotation around the Z-axis, in radians,
ext_collis – parameters for setting external collision protection for a given robot body
ext_collis = [body, flag], where:
  body – the name of the body {"All", "Move", "Arms", "LArm", "RArm"}
  flag – TRUE when the body external collision protection has to be enabled,
walk_arms – parameters for enabling robot arms while walking
walk_arms = [left, right], where:
  left – TRUE when the left arm has to be enabled,
  right – TRUE when the right arm has to be enabled,
foot_prot – a boolean flag for setting foot contact protection. If the flag is TRUE then the protection is enabled,
stiffness – name or names of joints for which stiffness will be set,
stiffness = [joints, values], where:
  joints – a name or names of joints,
  values – stiffness value in the range of 0.0 and 1.0,
dpost – arguments for posture interpolation (memorized argument of the POSTURE command),
dpost = [posture, speed], where:
  posture – a name of a predefined posture to be attained,
  speed – relative speed between 0.0 and 1.0,
dja – desired joint angles with parameters (memorized argument of the INTERPOLATION command)],
dja = [joints, values, fractionMaxSpeed], where:
  joints – a name or names of joints,
  values – one or more angles in radians,
  fractionMaxSpeed – fraction of the maximum speed during the interpolation of angles,

- Proprioceptive input from the real effector ${}^{E}_{x}e_{\text{core,body}}$:

[3] NAOqi software controls NAO at 50 Hz, i.e. the control period is 20 ms, thus the motion increment must be defined for that period – we call it real effector control step. Thus the sampling period of this virtual effector is 20 ms.

attained – returns TRUE if the predefined posture or desired position was attained,
cpost – returns current robot posture,
cja – current joint angles,
cja = [joints, values], where:
- joints – a name or names of joints,
- values – one or more angles in radians,

odm – current robot position expressed in Cartesian coordinates,
odm = [x, y], where:
- x – current robot position with respect to the X-axis of the global coordinate system, in meters,
- y – current robot position with respect to the Y-axis of the global coordinate system, in meters,

cmp – current motor positions,
value – current value of a parameter named as ${}^{c}_{x}e_{\text{core,body}}$[data_name]; this value is transmitted to the control subsystem in response to its querry
im – current inertial measurement unit (IMU) data,
im = $[\theta, \omega]$, where:
- θ – current robot orientation angle around the Z-axis in radians,
- ω – velocity around the Z-axis, in radians per second,

- Input from the control subsystem ${}^{c}_{x}e_{\text{core,body}}$:

cmd – command from the control subsystem,
$\text{cmd} \in \{\text{MOVE}, \text{STOP}, \text{POSTURE}, \text{INTERPOLATION}, \text{GET}\}$,

arg – arguments from the control subsystem;
$\text{arg} = [\text{velocity}, \text{dpose}, \text{dpost}, \text{dja}, \text{data\_name}]$, where:

- velocity – velocity of motion with respect to the robot coordinate frame (memorized argument of the MOVE command);
  $\text{velocity} = [\text{v}_\text{x}, \text{v}_\text{y}, \omega]$, where:
  - v_x – velocity along the X-axis, in meters per second,
  - v_y – velocity along the Y-axis, in meters per second,
  - ω – velocity around the Z-axis, in radians per second,
- dpose – desired position with respect to the robot coordinate frame (memorized argument of the MOVE command);
  $\text{dpose} = [\text{x}, \text{y}, \theta]$, where:
  - x – distance along the X-axis, in meters,
  - y – distance along the Y-axis, in meters,
  - θ – rotation around the Z-axis, in radians,
- dpost – arguments for posture interpolation (memorized argument of the POSTURE command),
  $\text{dpost} = [\text{posture}, \text{speed}]$, where:
  - posture – name of a predefined posture to be attained,
  - speed – relative speed between 0.0 and 1.0,
- dja – desired joint angles with parameters (memorized argument of the INTERPOLATION command)],
  $\text{dja} = [\text{joints}, \text{values}, \text{fractionMaxSpeed}]$, where:
  - joints – a name or names of joints,
  - values – one or more angles in radians,
  - fractionMaxSpeed – fraction of the maximum speed during the interpolation of angles,
- stiffness – name or names of joints for which stiffness will be set,
  $\text{stiffness} = [\text{joints}, \text{values}]$, where:
  - joints – a name or names of joints,
  - value – stiffness value in the range of 0.0 and 1.0,
- data_name – is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module

- Proprioceptive output to the control subsystem ${}^{c}_{y}e_{\text{core,body}}$:

attained – TRUE if the predefined posture, desired interpolation or position was attained or executed,

value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

pose – current robot position estimated by Extended Kalman Filter (EKF),
$\text{pose} = [\text{x}, \text{y}, \theta]$, where:

- x – current robot position in X-axis in Cartesian coordinate system in meters,
- y – current robot position in Y-axis in Cartesian coordinate system in meters,
- θ – current angle rotated around Z-axis in radians.

3.1.2 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,idle}}$ of the virtual effector $e_{\text{core,body}}$

- Transition function:

$$ {}^{e,e}f_{\text{core,body,idle}} \triangleq {}^{e}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big) $$

$$ {}^{e,c}f_{\text{core,body,idle}} \triangleq {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big) $$

This transition function estimates the current robot position based on the data received from the real effector $E_{\text{core,body}}$. The EKF function is an Extended Kalman Filter (EKF).

- Terminal condition

$$ \begin{aligned} {}^{e}f^{\tau}_{\text{core,body,idle}} \triangleq\ & ({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{MOVE}) \vee ({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{POSTURE}) \vee \\ & ({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{INTERPOLATION}) \vee ({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{GET}) \vee \\ & ({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STIFFNESS}) \vee ({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{MOVE\_TO}) \end{aligned} $$

When one of the above mentioned commands is obtained from the control subsystem the virtual effector stops being idle and immediately commences with the commanded motion.

3.1.3 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,move}}$ of the virtual effector $e_{\text{core,body}}$

- Transition functions:

$$ {}^{e,E}f_{\text{core,body,move}} \triangleq $$
$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] & = & \text{MOVE} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{velocity}] & = & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{arg}[\text{velocity}]] \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{ext\_collis}] & = & [\text{"All"}, \text{FALSE}] \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{walk\_arms}] & = & [\text{TRUE}, \text{TRUE}] \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{foot\_prot}] & = & \text{TRUE} \end{array} \right\} & \text{for } i = i_0 \\ {}^{E}_{y}e^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] \neq \text{STOP} \\ \left\{ \begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] & = & \text{MOVE} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{velocity}] & = & 0 \end{array} \right\} & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \end{cases} $$

$$ {}^{e,e}f_{\text{core,body,move}} \triangleq $$
$$ \begin{cases} {}^{e}e^{i+1}_{\text{core,body}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] \neq \text{STOP} \\ \text{TRUE} & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \end{cases} \\ {}^{e}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big) \end{cases} $$

$$ {}^{e,c}f_{\text{core,body,move}} \triangleq {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big) $$

This transition function transfers to the real effector $E_{\mathrm{core,body}}$ the command and the velocity at which the body should move and subsequently monitors the input from the control subsystem c_{core}. The EKF function estimates the current robot position using the Extended Kalman Filter (EKF). Once the STOP command is obtained from the control subsystem the commanded velocity, for the real effector, is reset to zero. Moreover, the estimated current robot position is transferred to the control subsystem $c_{\mathrm{core,body}}$. If the STOP command is not delivered the robot will, in principle, move endlessly with the prescribed velocity. The NAOqi move function was used in the activity of behaviour ${}^{e}_{+}\mathcal{B}_{\mathrm{core,body,move}}$.

- Terminal condition:

$$ {}^{e}f^{\tau}_{\mathrm{core,body,move}} \triangleq {}^{e}e^{i}_{\mathrm{core,body}}[\mathrm{tm}] = \mathrm{TRUE} $$

The STOP command obtained from the control subsystem switches the termination marker ${}^{e}e^{i}_{\mathrm{core,body}}[\mathrm{tm}]$ and this, in the next, control step causes the termination of the motion.

3.1.4 Behaviour ${}^{e}_{+}\mathcal{B}_{\mathrm{core,body,moveto}}$ of the virtual effector $e_{\mathrm{core,body}}$

- Transition functions:

$$ {}^{e,E}f_{\mathrm{core,body,moveto}} \triangleq $$
$$ \begin{cases} \left.\begin{cases} {}^{E}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{cmd}] & = \mathrm{MOVE\_TO} \\ {}^{E}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{dpose}] & = {}^{c}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{arg}[\mathrm{dpose}]] \\ {}^{E}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{ext\_collis}] & = [\text{"All"}, \mathrm{FALSE}] \\ {}^{E}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{walk\_arms}] & = [\mathrm{TRUE}, \mathrm{TRUE}] \\ {}^{E}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{foot\_prot}] & = \mathrm{TRUE} \end{cases}\right\} & \text{for } i = i_0 \\ {}^{E}_{y}e^{i+1}_{\mathrm{core,body}} = - & \text{for } i \neq i_0 \wedge \\ & {}^{c}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{cmd}] \neq \mathrm{STOP} \\ {}^{E}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{cmd}] = \mathrm{STOP\_MOVE} & \text{for } i \neq i_0 \wedge \\ & {}^{c}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{cmd}] = \mathrm{STOP} \end{cases} $$

$$ {}^{e,e}f_{\mathrm{core,body,moveto}} \triangleq $$
$$ \begin{cases} {}^{e}e^{i+1}_{\mathrm{core,body}}[\mathrm{tm}] = \begin{cases} \mathrm{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{cmd}] \neq \mathrm{STOP} \\ \mathrm{TRUE} & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{cmd}] = \mathrm{STOP} \vee \\ & {}^{E}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{attained}] = \mathrm{TRUE} \end{cases} \\ {}^{e}e^{i+1}_{\mathrm{core,body}}[\mathrm{pose}] = \mathrm{EKF}\left({}^{E}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{odm}], {}^{E}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{im}], {}^{e}e^{i}_{\mathrm{core,body}}[\mathrm{pose}] \right) \end{cases} $$

$$ {}^{e,c}f_{\mathrm{core,body,moveto}} \triangleq {}^{c}_{y}e^{i+1}_{\mathrm{core,body}}[\mathrm{pose}] = \mathrm{EKF}\left({}^{E}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{odm}], {}^{E}_{x}e^{i}_{\mathrm{core,body}}[\mathrm{im}], {}^{e}e^{i}_{\mathrm{core,body}}[\mathrm{pose}] \right) $$

This transition function transfers to the real effector $E_{\mathrm{core,body}}$ the command and the desired robot position with respect to the robot coordinate frame and subsequently monitors the input from the control subsystem c_{core}. The EKF function estimates the current robot position using the Extended Kalman Filter (EKF). Once the STOP command is obtained from the control subsystem the movement is terminated. Moreover, the estimated current robot position is transferred to the control subsystem $c_{\text{core,body}}$. The NAOqi moveto function was used in the activity of behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,moveto}}$.

- Terminal condition:

$$ {}^{e}f^{\tau}_{\text{core,body,moveto}} \triangleq {}^{e}e^{i}_{\text{core,body}}[\text{tm}] = \text{TRUE} $$

The STOP command obtained from the control subsystem switches the termination marker ${}^{e}e^{i}_{\text{core,body}}[\text{tm}]$ and this, in the next, control step causes the termination of the motion.

3.1.5 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,posture}}$ of the virtual effector $e_{\text{core,body}}$

- Transition function:

$$ {}^{e,E}f_{\text{core,body,posture}} \triangleq $$
$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] & = & \text{GO\_TO\_POSTURE} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{dpost}] & = & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{arg}[\text{dpost}]] \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{stiffness}] & = & [\text{"Body"}, 1.0] \end{array} \right\} & \text{for } i = i_0 \\ {}^{E}_{y}e^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0 \wedge \\ & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] \neq \text{STOP} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] \;=\; \text{STOP\_MOVE} & \text{for } i \neq i_0 \wedge \\ & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \end{cases} $$

$$ {}^{e,c}f_{\text{core,body,posture}} \triangleq $$
$$ \begin{cases} {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{attained}] = \begin{cases} {}^{E}_{x}e^{i}_{\text{core,body}}[\text{attained}] & \text{for new}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{attained}]\right) \\ \text{FALSE} & \text{for } {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \\ - & \text{otherwise} \end{cases} \\ {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}]\right) \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.

$$ {}^{e,e}f_{\text{core,body,posture}} \triangleq $$
$$ \begin{cases} {}^{e}e^{i+1}_{\text{core,body}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] \neq \text{STOP} \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \left({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \vee \right. \\ & \left. \text{new}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{attained}]\right)\right) \end{cases} \\ {}^{e}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}]\right) \end{cases} $$

This transition function transfers to the real effector $E_{\text{core,body}}$ the predefined posture to be attained with the relative speed. Once the STOP command is obtained from the control subsystem the commanded posture interpolation is terminated. The EKF function estimates the current robot position using the Extended Kalman Filter (EKF). The NAOqi goToPosture function was used in the activity of the behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,posture}}$. Upon termination of this behaviour the real effector $E_{\text{core,body}}$ transfers to the virtual effector $e_{\text{core,body}}$ the information whether the posture was attained. This information and the estimated current robot position by EKF are further transferred to the control subsystem $c_{\text{core,body}}$.

- Terminal condition:

$$ {}^{e}f^{\tau}_{\text{core,body,posture}} \triangleq {}^{e}e^{i}_{\text{core,body}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{e}e^{i}_{\text{core,body}}[\text{tm}]$ is switched when the STOP command is obtained from the control subsystem or the information about the posture interpolation is obtained from the real effector $E_{\text{core,body}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.1.6 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,inter}}$ of the virtual effector $e_{\text{core,body}}$

- Transition function:

$$ {}^{e,E}f_{\text{core,body,inter}} \triangleq $$

$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] & = & \text{SET\_ANGLES} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{dja}] & = & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{arg}[\text{dja}]] \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{stiffness}] & = & [\text{dja}, 1.0] \end{array} \right\} & \text{for } i = i_0 \\ {}^{E}_{y}e^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0 \wedge \\ & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] \neq \text{STOP} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] \;=\; \text{STOP\_MOVE} & \text{for } i \neq i_0 \wedge \\ & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \end{cases} $$

$$ {}^{e,c}f_{\text{core,body,inter}} \triangleq $$

$$ \begin{cases} {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{attained}] = \begin{cases} \text{TRUE} & \text{for } {}^{e}e^{i}_{\text{core,body}}[\text{dja}] = {}^{E}_{x}e^{i}_{\text{core,body}}[\text{cja}] \\ \text{FALSE} & \text{for } {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \\ - & \text{otherwise} \end{cases} \\ {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \right) \end{cases} $$

$$
{}^{e,e}f_{\text{core,body,inter}} \triangleq
\begin{cases}
{}^{e}e^{i+1}_{\text{core,body}}[\text{tm}] = \begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \\
 & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] \neq \text{STOP} \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \\
 & \Big({}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}] = \text{STOP} \vee \\
 & {}^{e}e^{i}_{\text{core,body}}[\text{dja}] = {}^{E}_{x}e^{i}_{\text{core,body}}[\text{cja}] \Big) \vee \\
 & \text{new}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{attained}] \Big) \Big)
\end{cases} \\
{}^{e}e^{i+1}_{\text{core,body}}[\text{dja}] \; = \; {}^{c}_{x}e^{i}_{\text{core,body}}[\text{arg}[\text{dja}]] \qquad \text{for } i = i_0 \\
{}^{e}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big)
\end{cases}
$$

This transition function transfers to the real effector $E_{\text{core,body}}$ the vector of joint angles and the fraction of maximum speed. The NAOqi setAngles function was used in the activity of the behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,inter}}$. The EKF function estimates the current robot position using the Extended Kalman Filter (EKF). Once the STOP command is obtained from the control subsystem the commanded joint interpolation is terminated and the real effector $E_{\text{core,body}}$ returns to the virtual effector $e_{\text{core,body}}$ the information whether the posture was attained. This information and the estimated current robot position are transferred to the control subsystem $c_{\text{core,body}}$.

- Terminal condition:

$$
{}^{e}f^{\tau}_{\text{core,body,inter}} \triangleq {}^{e}e^{i}_{\text{core,body}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{e}e^{i}_{\text{core,body}}[\text{tm}]$ is switched when the STOP command is obtained from the control subsystem or the desired joint angles are attained. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.[4]

3.1.7 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,set\_stiffness}}$ of the virtual effector $e_{\text{core,body}}$

- Transition function:

$$
{}^{e,E}f_{\text{core,body,stiffness}} \triangleq
\begin{cases}
{}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] & = \text{STIFFNESS\_INTERPOL} \\
{}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{stiffness}] & = {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{stiffness}]]
\end{cases}
$$

$$
\begin{aligned}
{}^{e,e}f_{\text{core,body,stiffness}} \triangleq \; & {}^{e}e^{i+1}_{\text{core,body}}[\text{pose}] = \\
& \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big)
\end{aligned}
$$

$$
{}^{e,c}f_{\text{core,body,stiffness}} \triangleq {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\Big({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}] \Big)
$$

[4]The equality of current and desired position should be treated approximately – as in any system taking into account noisy measurements.

This transition function transfers to the real effector $E_{\text{core,body}}$ the stiffness parameters to be set for the given joints and using Extended Kalman Filter (EKF) estimates the current robot position. The NAOqi stiffnessInterpolation function was used in the activity of the behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,stiffness}}$.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,body,stiffness}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.1.8 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,body,get}}$ of the virtual effector $e_{\text{core,body}}$

- Transition function:

$$ {}^{e,c}f_{\text{core,body,get}} \triangleq \begin{cases} {}^{c}_{y}e^{i+1}_{\text{core,body}} = - & \text{for } \neg\text{new}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{value}]\right) \\ {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{value}] = {}^{E}_{x}e^{i}_{\text{core,body}}[\text{value}] & \text{for new}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{value}]\right) \\ {}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}]\right) & \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.

$$ {}^{e,E}f_{\text{core,body,get}} \triangleq $$
$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{cmd}] & = & \text{GET\_DATA} \\ {}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{data\_name}] & = & {}^{c}_{x}e^{i}_{\text{core,body}}[\text{arg}[\text{data\_name}]] \end{array} \right\} & \text{for } i = i_0 \\ {}^{E}_{y}e^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{e,e}f_{\text{core,body,get}} \triangleq $$
$$ \begin{cases} {}^{e}e^{i+1}_{\text{core,body}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{value}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{value}]\right) \end{cases} \\ {}^{e}e^{i+1}_{\text{core,body}}[\text{pose}] = \text{EKF}\left({}^{E}_{x}e^{i}_{\text{core,body}}[\text{odm}], {}^{E}_{x}e^{i}_{\text{core,body}}[\text{im}], {}^{e}e^{i}_{\text{core,body}}[\text{pose}]\right) \end{cases} $$

The transition function returns to the control subsystem ${}^{c}_{y}e_{\text{core,body}}$ the current value of a parameter ${}^{c}_{y}e^{i+1}_{\text{core,body}}[\text{value}]$ named as data_name. It estimates using Extended Kalman Filter (EKF) the current robot position.

- Terminal condition:

$$ {}^{e}f^{\tau}_{\text{core,body,get}} \triangleq {}^{e}e^{i}_{\text{core,body}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{e}e^{i}_{\text{core,body}}[\text{tm}]$ is switched when the value of the desired parameter is obtained from the real effector $E_{\text{core,body}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

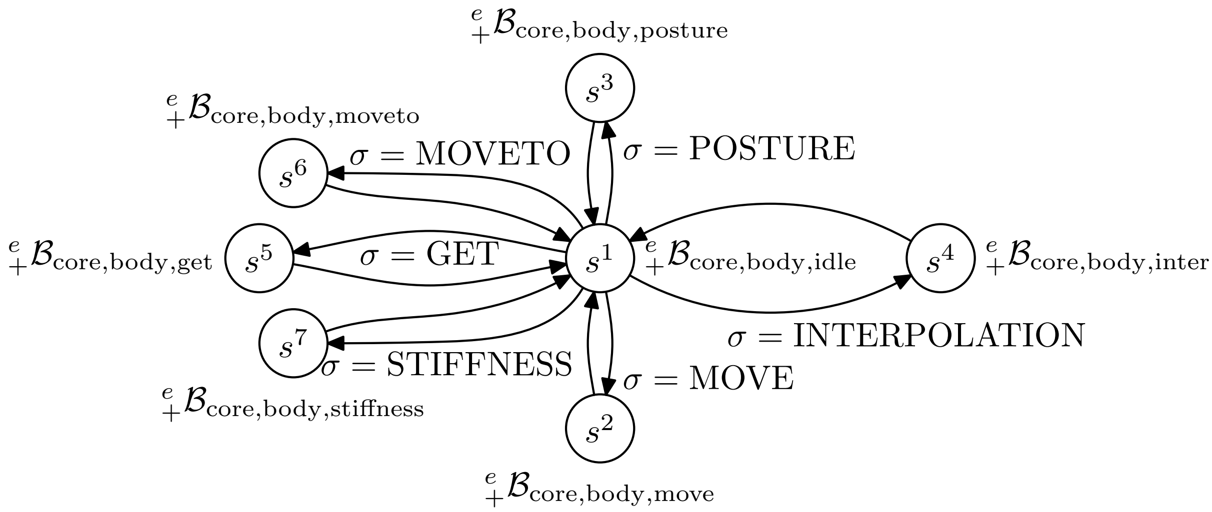

Figure 5: FSM governing the activities of the body virtual effector $e_{\text{core,body}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}e^{i}_{\text{core,body}}[\text{cmd}]$

3.1.9 FSM governing the virtual effector $e_{\text{core,body}}$

The seven state automaton (FSM) governing the activities of the body virtual effector $e_{\text{core,body}}$ is presented in fig. 5.

3.2 Virtual effector $e_{\text{core,ls}}$

The virtual effector $e_{\text{core,ls}}$ controls the loudspeaker.

3.2.1 Communication buffers and internal memory of the virtual effector $e_{\text{core,ls}}$

- Internal memory ${}^{e}e_{\text{core,ls}}$:
  tm – termination marker

- Real effector control ${}^{E}_{y}e_{\text{core,ls}}$:
  cmd – command from the virtual effector,
  text – current text to synthesize,
  data_name – data name of a parameter stored in a NAOqi ALMemory module,
  fp – the path to the file that will be reproduced,
  begin_position – position in second where the playing should begin,
  params – loudspeaker virtual effector parameters,
  $\text{params} = [\text{dvt}, \text{dl}, \text{dv}, \text{spr}]$, where:
    dvt – desired voice type,
    dl – desired language,
    dv – desired volume,
    spr – stereo panorama requested (-1.0 : left, 1.0 : right, 0.0 : center),

- Proprioceptive input from the real effector ${}^{E}_{x}e_{\text{core,ls}}$:
  synthesized – returns TRUE when the current text synthesis is finished
  value – current value of a parameter named as ${}^{E}_{y}e^{i+1}_{\text{core,body}}[\text{data\_name}]$

- Input from the control subsystem ${}^{c}_{x}e_{\text{core,ls}}$:

cmd – command from the control subsystem, cmd $\in \{\text{SAY}, \text{PLAY}, \text{STOP}, \text{GET}\}$,
arg – arguments from the control subsystem,
arg = [text, fp, params, data_name, volume, spr, playLoop, begin_position], where:

text – the text to be transformed into the synthesized sound,
fp – the path to the file that will be reproduced,
params – loudspeaker virtual effector parameters, params = [dvt, dl, dv, spr], where:

dvt – desired voice type,
dl – desired language,
dv – desired volume,
spr – stereo panorama requested (-1.0 : left, 1.0 : right, 0.0 : center),

data_name – data_name is a string that contains a name of a parameter stored in a NAOqi ALMemory module, it is used to query the value of the NAOqi parameter stored in the AlMemory module,
playLoop – plays a file in a loop if the flag is set to TRUE, otherwise plays once,
begin_position – position in second where the playing should begin,

- Proprioceptive output to the control subsystem ${}^{c}_{y}e_{\text{core,ls}}$:

reply – information about the termination of sound synthesis,
value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

3.2.2 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,ls,idle}}$ of the virtual effector $e_{\text{core,ls}}$

- Transition function:

$$ {}^{e}f_{\text{core,ls,idle}} \triangleq {}_{y}e^{i+1}_{\text{core,ls}} = - $$

- Terminal condition:

$$ \begin{aligned} {}^{e}f^{\tau}_{\text{core,ls,idle}} \triangleq\; & ({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{SAY}) \vee ({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{PLAY}) \vee ({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{GET}) \\ & \vee({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{SET\_PARAMS}) \vee ({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{STOP}) \end{aligned} $$

When one of the above mentioned commands is obtained from the control subsystem the virtual effector stops being idle to immediately commence with the commanded behaviour.

3.2.3 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,ls,say}}$ of the virtual effector $e_{\text{core,ls}}$

- Transition function:

$$
{}^{e,E}f_{\text{core,ls,say}} \triangleq
\begin{cases}
\left\{
\begin{array}{lcl}
{}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] & = & \text{SAY} \\
{}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{text}] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{text}]] \\
{}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{params}[\text{dl}]] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{params}[\text{dl}]]]
\end{array}
\right\} & \text{for } i = i_0 \\
{}^{E}_{y}e^{i+1}_{\text{core,ls}} = - & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] \neq \text{STOP} \\
{}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] = \text{STOP\_ALL} & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{STOP}
\end{cases}
$$

$$
{}^{e,e}f_{\text{core,ls,say}} \triangleq {}^{e}e^{i+1}_{\text{core,ls}}[\text{tm}] =
\begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] \neq \text{STOP} \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \Big({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{STOP} \vee \\
& \quad {}^{E}_{x}e^{i}_{\text{core,ls}}[\text{synthesized}] = \text{TRUE} \Big)
\end{cases}
$$

$$
{}^{e,c}f_{\text{core,ls,say}} \triangleq {}^{c}_{y}e^{i+1}_{\text{core,ls}}[\text{reply}] =
$$
$$
\begin{cases}
\text{TRUE} & \text{when } \Big({}^{E}_{x}e^{i}_{\text{core,ls}}[\text{synthesized}] = \text{TRUE} \Big) \vee \Big({}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}] = \text{STOP} \Big) \\
- & \text{otherwise}
\end{cases}
$$

These transition functions transfer to the real effector $E_{\text{core,ls}}$ the text to synthesize and subsequently monitor the input from the control subsystem c_{core}. Once the STOP command is obtained from the control subsystem the commanded text synthesis is interrupted. If the STOP command is not delivered the sound will be, in principle, synthesized until the information about synthesis termination will be obtained from the real effector $E_{\text{core,ls}}$. It should be noted that the activities performed by the NAOqi say or stopAll functions are executed within the real effector $E_{\text{core,ls}}$.

- Terminal condition:

$$
{}^{e}f^{\tau}_{\text{core,ls,say}} \triangleq {}^{e}e^{i}_{\text{core,ls}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{e}e^{i}_{\text{core,ls}}[\text{tm}]$ is switched when the STOP command is obtained from the control subsystem or the information about synthesis termination is delivered by the real effector $E_{\text{core,ls}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.2.4 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,ls,play}}$ of the virtual effector $e_{\text{core,ls}}$

- Transition function:

$$ {}^{e,E}f_{\text{core,ls,play}} \triangleq \begin{cases} \left.\begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] & = & \text{PLAY\_FILE\_IN\_LOOP} \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{fp}] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{fp}]] \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{params}[\text{dv}]] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{params}[\text{dv}]]] \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{params}[\text{spr}]] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{params}[\text{spr}]]] \end{array}\right\} & \text{for playLoop} = \text{TRUE} \\ \left.\begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] & = & \text{PLAY\_FROM\_POSITION} \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{fp}] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{fp}]] \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{params}[\text{dv}]] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{params}[\text{dv}]]] \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{params}[\text{spr}]] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{params}[\text{spr}]]] \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{begin\_position}] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{begin\_position}]] \end{array}\right\} & \text{for playLoop} = \text{FALSE} \end{cases} $$

These transition functions transfer to the real effector $E_{\text{core,ls}}$ the command, path to the file that will be played, volume and the balance. It should be noted that all the activities performed by the NAOqi playFileInLoop or playFileFromPosition function are executed within the real effector $E_{\text{core,ls}}$.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,ls,play}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.2.5 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,ls,stop}}$ of the virtual effector $e_{\text{core,ls}}$

- Transition function:

$$ {}^{e,E}f_{\text{core,ls,stop}} \triangleq {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] = \text{STOP\_ALL} $$

These transition function transfer to the real effector $E_{\text{core,ls}}$ the command to stop sound synthesis. It should be noted that the activity performed by the NAOqi stopAll function is executed within the real effector $E_{\text{core,ls}}$.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,ls,stop}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.2.6 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,ls,set\_params}}$ of the virtual effector $e_{\text{core,ls}}$

- Transition function:

$$ {}^{e,E}f_{\text{core,ls,set\_params}} \triangleq \begin{cases} {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] & = \text{SET\_PARAMETERS} \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{params}] & = {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{params}]] \end{cases} $$

This transition function transfers to the real effector $E_{\text{core,ls}}$ the parameters obtained from the control subsystem. It should be noted that all the activities performed by the NAOqi setParameter and other functions are executed within the real effector $E_{\text{core,ls}}$.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,ls,set\_params}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.2.7 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,ls,get}}$ of the virtual effector $e_{\text{core,ls}}$

- Transition function:

$$ {}^{e,c}f_{\text{core,ls,get}} \triangleq \begin{cases} {}^{c}_{y}e^{i+1}_{\text{core,ls}} = - & \text{for } \neg\text{new}\left({}^{E}_{x}e^{i}_{\text{core,ls}}[\text{value}]\right) \\ {}^{c}_{y}e^{i+1}_{\text{core,ls}}[\text{value}] = {}^{E}_{x}e^{i}_{\text{core,ls}}[\text{value}] & \text{for new}\left({}^{E}_{x}e^{i}_{\text{core,ls}}[\text{value}]\right) \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.
${}^{e,E}f_{\text{core,ls,get}} \triangleq$

$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{cmd}] & = & \text{GET\_DATA} \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}}[\text{data\_name}] & = & {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{arg}[\text{data\_name}]] \end{array} \right\} & \text{for } i = i_0 \\ {}^{E}_{y}e^{i+1}_{\text{core,ls}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{e,e}f_{\text{core,ls,get}} \triangleq {}^{e}e^{i+1}_{\text{core,ls}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{E}_{x}e^{i}_{\text{core,ls}}[\text{value}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{E}_{x}e^{i}_{\text{core,ls}}[\text{value}]\right) \end{cases} $$

The transition function returns to the control subsystem ${}^{c}_{y}e_{\text{core,ls}}$ the current value of the parameter ${}^{c}_{y}e^{i+1}_{\text{core,ls}}[\text{value}]$ named as data_name.

- Terminal condition:

$$ {}^{e}f^{\tau}_{\text{core,ls,get}} \triangleq {}^{e}e^{i}_{\text{core,ls}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{e}e^{i}_{\text{core,ls}}[\text{tm}]$ is switched when the value of a desired parameter is obtained from the real effector $E_{\text{core,ls}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.2.8 FSM governing the virtual effector $e_{\text{core,ls}}$

The five state automaton (FSM) governing the activities of the loudspeaker virtual effector $e_{\text{core,ls}}$ is presented in fig. 11.

3.3 Virtual receptor $r_{\text{core,mic}}$

The virtual receptor $r_{\text{core,mic}}$ is responsible for acquiring and aggregating data obtained by the microphones. It should be noted that all the activities performed by the NAOqi functions are executed within the real receptor, thus NAOqi library is treated as an element of the real receptor $R_{\text{core,mic}}$. Below the contents of transmission buffers as well as the transition functions and terminal conditions of the behaviours of the virtual receptor are defined. A dash in the definition of a transition function implies that a certain output variable or a certain set of output variables is not assigned a value. This further implies that the values

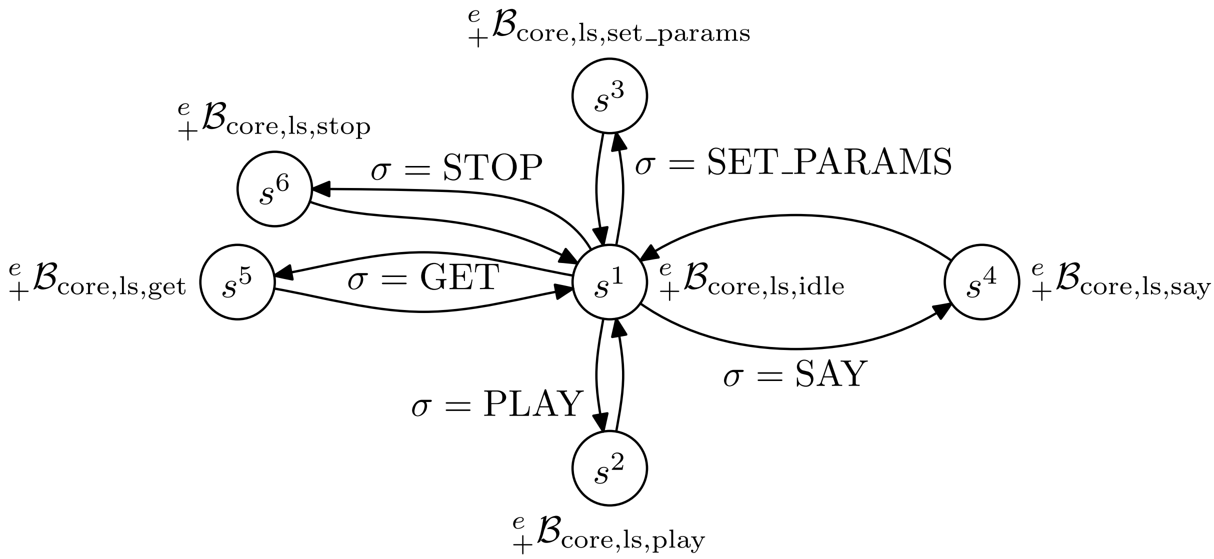

Figure 6: FSM governing the activities of the body loudspeaker effector $e_{\text{core,ls}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}e^{i}_{\text{core,ls}}[\text{cmd}]$

of those variables are not sent to an associated subsystem. Usually such variables are not mentioned in the definition of the transfer function, but if they appear in some iterations of the behaviour and in some they do not, for the purpose of completeness the lack of value assignment is signalled by a dash. Similarly, if a certain behaviour is based on a transition function that produces no output values, this is signalled by a dash, otherwise one could come to a false conclusion that there exist behaviours not based on transition functions.

3.3.1 Communication buffers and internal memory of the virtual receptor $r_{\text{core,mic}}$

- Internal memory ${}^{r}r_{\text{core,mic}}$:

  rd – contains raw data from the microphones,
  tm – termination marker,
  silence – maximum number of iterations during which silence in speech is acceptable: ${}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{time}]]$ or ${}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{st}]]$,
  cit – counter (iterator),

- Real receptor control ${}^{R}_{y}r_{\text{core,mic}}$:

  rec – TRUE when the recoding is to be triggered, FALSE when it is to be stopped,
  cmd – command from the virtual receptor,
  data_name – data name of a parameter stored in a NAOqi ALMemory module,
  fp – file containing the recorded signal samples,
  threshold – threshold required for word recognition,

- Input from the real receptor ${}^{R}_{x}r_{\text{core,mic}}$:

  enrg – contains signal energy for each microphone,
  value – current value of a parameter named as ${}^{c}_{x}r_{\text{core,touch}}[\text{data\_name}]$; this value is transmitted to the control subsystem in response to its query
  rw – contains a list of pairs, $\text{rw} = [(\text{recog\_word}, \text{recog\_prob}), \ldots]$, where:
    recog_word – recognized word,
    recog_prob – probability of correct speech recognition for recog_word,

- Input from the control subsystem ${}^{c}_{x}r_{\text{core,mic}}$:

  cmd – command from the control subsystem, $\text{cmd} \in \{\text{RECORD}, \text{REGISTER}, \text{ABORT}, \text{RECOGNIZE}, \text{GET}\}$,

  arg – arguments from control subsystem, $\text{arg} = [\text{params}, \text{dct}, \text{data\_name}, \text{fp}, \text{sampling\_period}, \text{time}, \text{enrg}, \text{st}]$, where:

params	–	contains microphone parameters such as: sample rate and microphone channels,
dct	–	contains the list of words that should be recognized,
data_name	–	is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module
fp	–	file containing the recorded signal samples,
sampling_period	–	sampling period of microphone virtual receptor $r_{\text{core,mic}}$,
time	–	commanded duration of the recording,
enrg	–	threshold of signal energy for microphones,
st	–	time during which the microphone signal energy is compared with the threshold. When during this time the signal will be lower than the threshold then the recording stops,
threshold	–	threshold required for word recognition,

- Output produced by the virtual receptor for the control subsystem ${}^{c}_{y}r_{\text{core,mic}}$:

  rec – TRUE when the recoding is terminated,

  rw – recognized word with the biggest probability of word recognition,

  value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

3.3.2 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,mic,idle}}$ of the virtual receptor $r_{\text{core,mic}}$

- Transition function:

$$ {}^{r}f_{\text{core,mic,idle}} \triangleq {}_{y}r^{i+1}_{\text{core,mic}} = - $$

- Terminal condition:

$$ \begin{aligned} {}^{r}f^{\tau}_{\text{core,mic,idle}} \triangleq\ & ({}^{c}_{x}r^{i}_{\text{core,mic}}[\text{cmd}] = \text{RECORD}) \vee ({}^{c}_{x}r^{i}_{\text{core,mic}}[\text{cmd}] = \text{REGISTER}) \vee \\ & ({}^{c}_{x}r^{i}_{\text{core,mic}}[\text{cmd}] = \text{GET}) \vee ({}^{c}_{x}r^{i}_{\text{core,mic}}[\text{cmd}] = \text{RECOGNIZE}) \end{aligned} $$

When any of the above mentioned commands is obtained from the control subsystem the virtual receptor stops being idle and immediately transfers to a state in which it gathers sound samples.

3.3.3 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,mic,record}}$ of the virtual receptor $r_{\text{core,mic}}$

- Transition functions:

$$
{}^{r,R}f_{\text{core,mic,record}} \triangleq
\begin{cases}
\left\{ \begin{array}{lcl} {}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] & = & \text{START\_RECORDING} \\ {}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{fp}] & = & {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{fp}]] \end{array} \right\} & \text{for } i = i_0 \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0 \wedge i < i_0 + {}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}] \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] = \text{STOP\_RECORDING} & \text{for } i = i_0 + {}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}]
\end{cases}
$$

$$
{}^{r,r}f_{\text{core,mic,record}} \triangleq
\begin{cases}
{}^{r}r^{i+1}_{\text{core,mic}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge i < i_0 + {}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}] \\ \text{TRUE} & \text{for } i = i_0 + {}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}] \end{cases} \\
{}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}] = {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{time}]] / {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{sampling\_period}]] \quad \text{for } i = i_0
\end{cases}
$$

$$
{}^{r,c}f_{\text{core,mic,record}} \triangleq {}^{c}_{y}r^{i+1}_{\text{core,mic}}[\text{rec}] = \begin{cases} \text{TRUE} & \text{for } i = i_0 + {}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}] \\ - & \text{otherwise} \end{cases}
$$

These transition functions transfer to the real receptor $R_{\text{core,mic}}$ the command and the destination file path, where the sound will be recorded. The virtual receptor $r_{\text{core,mic}}$ initiates the recording and if in the speech a period of silence occurs then it waits at the most ${}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}]$ iterations and stops recording. The recording is done by the NAOqi function startMicrophonesRecording. When the time elapses then the virtual receptor calls the NAOqi function stopMicrophonesRecording and the recording is terminated.

- Terminal condition:

$$
{}^{r}f^{\tau}_{\text{core,mic,record}} \triangleq {}^{r}r^{i}_{\text{core,mic}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{r}r^{i}_{\text{core,mic}}[\text{tm}]$ is switched after ${}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}]$ iterations. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.3.4 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,mic,register}}$ of the virtual receptor $r_{\text{core,mic}}$

- Transition functions:

$$
{}^{r,R}f_{\text{core,mic,register}} \triangleq
\begin{cases}
\left.\begin{cases}
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] & = \text{START\_RECORDING} \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{fp}] & = {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{fp}]] \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{enable\_mic\_comp}] & = \text{TRUE} \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{params}] & = {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{params}]]
\end{cases}\right\} & \text{for } i = i_0 \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0 \wedge {}^{r}r^{i}_{\text{core,mic}}[\text{cit}] < {}^{r}r^{i}_{\text{core,mic}}[\text{silence}] \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] = \text{STOP\_RECORDING} & \text{for } i \neq i_0 \wedge {}^{r}r^{i}_{\text{core,mic}}[\text{cit}] = {}^{r}r^{i}_{\text{core,mic}}[\text{silence}]
\end{cases}
$$

$$
{}^{r,r}f_{\text{core,mic,register}} \triangleq
\begin{cases}
{}^{r}r^{i+1}_{\text{core,mic}}[\text{tm}] = \begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge {}^{r}r^{i}_{\text{core,mic}}[\text{cit}] < {}^{r}r^{i}_{\text{core,mic}}[\text{silence}] \\
\text{TRUE} & \text{for } {}^{r}r^{i}_{\text{core,mic}}[\text{cit}] = {}^{r}r^{i}_{\text{core,mic}}[\text{silence}]
\end{cases} \\
{}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}] = \\
{}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{st}]] / {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{sampling\_period}]] \quad \text{for } i = i_0 \\
{}^{r}r^{i+1}_{\text{core,mic}}[\text{cit}] = \begin{cases}
0 & \text{for } i = i_0 \vee {}^{R}_{x}r^{i}_{\text{core,mic}}[\text{enrg}] \geq {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{enrg}]] \\
{}^{r}r^{i}_{\text{core,mic}}[\text{cit}] + 1 & \text{for } {}^{R}_{x}r^{i}_{\text{core,mic}}[\text{enrg}] < {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{enrg}]]
\end{cases}
\end{cases}
$$

$$
{}^{r,c}f_{\text{core,mic,register}} \triangleq {}^{c}_{y}r^{i+1}_{\text{core,mic}}[\text{rec}] = \begin{cases}
\text{TRUE} & \text{for } {}^{r}r^{i}_{\text{core,mic}}[\text{cit}] = {}^{r}r^{i}_{\text{core,mic}}[\text{silence}] \\
- & \text{otherwise}
\end{cases}
$$

These transition functions transfer to the real receptor $R_{\text{core,mic}}$ the command, the destination file path, recording parameters and they additionally enable microphone computations. The virtual receptor $r_{\text{core,mic}}$ initiates the recording and waits ${}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}]$ iterations until the microphone energy measurement ${}^{R}_{x}r^{i}_{\text{core,mic}}[\text{enrg}]$ surpasses the threshold ${}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{enrg}]]$. If none of the measurements exceeds the threshold then the behaviour terminates. The recording is done by the NAOqi function subscribe. When the sampling time is exceeded then the virtual receptor calls the NAOqi function unsubscribe and then the recording is terminated.

- Terminal condition:

$$
{}^{r}f^{\tau}_{\text{core,mic,register}} \triangleq {}^{r}r^{i}_{\text{core,mic}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{r}r^{i}_{\text{core,mic}}[\text{tm}]$ is switched when during ${}^{r}r^{i+1}_{\text{core,mic}}[\text{silence}]$ iterations each microphone energy measurement ${}^{R}_{x}r^{i}_{\text{core,mic}}[\text{enrg}]$ does not exceed the threshold ${}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{enrg}]]$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.3.5 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,mic,recog}}$ of the virtual receptor $r_{\text{core,mic}}$

- Transition functions:

$$
{}^{r,R}f_{\text{core,mic,recog}} \triangleq
\begin{cases}
\begin{cases}
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] = \text{SUBSCRIBE} \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{dct}] = {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{dct}]]
\end{cases} & \text{for } i = i_0 \\
\begin{cases}
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] = \text{GET\_DATA} \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{data\_name}] = \text{LastWordRecognized}
\end{cases} & \text{for } i = i_0 + 1 \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] = \text{UNSUBSCRIBE} & \text{for } i > i_0 + 1 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{rw}]\right) \\
{}^{R}_{y}r^{i+1}_{\text{core,mic}} = - & \text{otherwise}
\end{cases}
$$

where new is a predicate being TRUE when a new value of its argument is obtained.

$$
{}^{r,r}f_{\text{core,mic,recog}} \triangleq
\begin{cases}
{}^{r}r^{i+1}_{\text{core,mic}}[\text{tm}] = \begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \\
\text{TRUE} & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{rw}]\right)
\end{cases} \\
{}^{r}r^{i+1}_{\text{core,mic}}[\text{threshold}] = {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{threshold}]] \qquad \text{for } i = i_0
\end{cases}
$$

$$
{}^{r,c}f_{\text{core,mic,recog}} \triangleq {}^{c}_{y}r^{i+1}_{\text{core,mic}}[\text{rw}] =
\begin{cases}
\text{fst}\left({}^{R}_{x}\text{r}^{\text{i}}_{\text{core,mic}}[\text{rw}]\right) & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{rw}]\right) \wedge \\
& \text{fst}\left({}^{R}_{x}\text{r}^{\text{i}}_{\text{core,mic}}[\text{rw}[\text{recog\_prob}]]\right) \geq \\
& {}^{r}r^{i}_{\text{core,mic}}[\text{threshold}] \\
\text{Empty} & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{rw}]\right) \wedge \\
& \text{fst}\left({}^{R}_{x}\text{r}^{\text{i}}_{\text{core,mic}}[\text{rw}[\text{recog\_prob}]]\right) < \\
& {}^{r}r^{i}_{\text{core,mic}}[\text{threshold}] \\
- & \text{otherwise}
\end{cases}
$$

These transition functions transfer to the real receptor $R_{\text{core,mic}}$ the recognition command (initiated by a subscription to a recognition module) and the dictionary with words to be recognized by the NAOqi functions. In the next iteration it sends to the real receptor the query to get the value of LastWordRecognized. If the subsequent iterations a new value appears in the buffer ${}^{R}_{x}r^{i}_{\text{core,mic}}[\text{rw}]$ then the virtual receptor sends to the real receptor a command to terminate the subscription. Additionally the transition function returns to the control subsystem ${}^{c}_{y}r_{\text{core,mic}}$ the recognized word ${}^{c}_{y}r^{i+1}_{\text{core,mic}}[\text{rw}]$.

- Terminal condition:

$$
{}^{r}f^{\tau}_{\text{core,mic,recog}} \triangleq {}^{r}r^{i}_{\text{core,mic}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{r}r^{i}_{\text{core,mic}}[\text{tm}]$ is switched when the new value appears in the buffer ${}^{R}_{x}r^{i}_{\text{core,mic}}[\text{rw}]$ from the real receptor $R_{\text{core,mic}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.3.6 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,mic,get}}$ of the virtual receptor $r_{\text{core,mic}}$

- Transition function:

$$
{}^{r,c}f_{\text{core,mic,get}} \triangleq \begin{cases} {}^{c}_{y}r^{i+1}_{\text{core,mic}} = - & \text{for } \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{value}]\right) \\ {}^{c}_{y}r^{i+1}_{\text{core,mic}}[\text{value}] = {}^{R}_{x}r^{i}_{\text{core,mic}}[\text{value}] & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{value}]\right) \end{cases}
$$

where new is a predicate being TRUE when a new value of its argument is obtained.
${}^{r,R}f_{\text{core,mic,get}} \triangleq$

$$
\begin{cases} \left.\begin{cases} {}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{cmd}] & = & \text{GET\_DATA} \\ {}^{R}_{y}r^{i+1}_{\text{core,mic}}[\text{data\_name}] & = & {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{arg}[\text{data\_name}]] \end{cases}\right\} & \text{for } i = i_0 \\ {}^{R}_{y}r^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0 \end{cases}
$$

$$
{}^{r,r}f_{\text{core,mic,get}} \triangleq {}^{r}r^{i+1}_{\text{core,mic}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{value}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,mic}}[\text{value}]\right) \end{cases}
$$

The transition function returns to the control subsystem ${}^{c}_{y}r_{\text{core,mic}}$ the current value of the parameter ${}^{c}_{y}r^{i+1}_{\text{core,mic}}[\text{value}]$ named as data_name.

- Terminal condition:

$$
{}^{r}f^{\tau}_{\text{core,mic,get}} \triangleq {}^{r}r^{i}_{\text{core,mic}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{r}r^{i}_{\text{core,mic}}[\text{tm}]$ is switched when the value of a desired parameter is obtained from the real receptor $R_{\text{core,mic}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.3.7 FSM governing the virtual receptor $r_{\text{core,mic}}$

The five state automaton (FSM) governing the activities of the microphone virtual receptor $r_{\text{core,mic}}$ is presented in fig. 7.

3.4 Virtual receptor $r_{\text{core,touch}}$

The virtual receptor $r_{\text{core,touch}}$ informs the control subsystem about the current status of its touch sensors.

3.4.1 Communication buffers and internal memory of the virtual receptor $r_{\text{core,touch}}$

- Internal memory ${}^{r}r_{\text{core,touch}}$:
  tm – termination marker

- Real receptor control ${}^{R}_{y}r_{\text{core,touch}}$:
  cmd – command from the virtual receptor,
  data_name – data name of a parameter stored in a NAOqi ALMemory module,

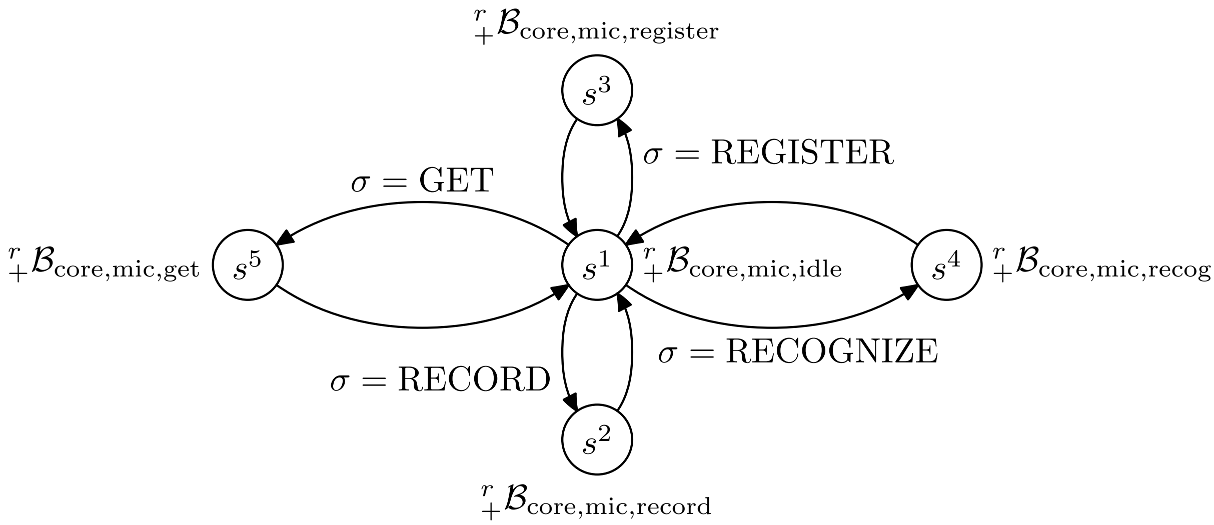

Figure 7: FSM governing the activities of the microphone virtual receptor $r_{\text{core,mic}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}r^{i}_{\text{core,mic}}[\text{cmd}]$

- Input from the real receptor ${}^{R}_{x}r_{\text{core,touch}}$:
  value – current value of a parameter named as ${}^{c}_{x}r_{\text{core,touch}}[\text{data\_name}]$; this value is transmitted to the control subsystem in response to its query

- Input from the control subsystem ${}^{c}_{x}r_{\text{core,touch}}$:
  cmd – command from the control subsystem, $\text{cmd} \in \{\text{GET}\}$,
  arg – arguments from the control subsystem; $\text{arg} = [\text{data\_name}]$, where:
  data_name – is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module

- Output produced by the virtual receptor for the control subsystem ${}^{c}_{y}r_{\text{core,touch}}$:
  value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

3.4.2 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,touch,idle}}$ of the virtual receptor $r_{\text{core,touch}}$

- Transition function:

$$ {}^{r}f_{\text{core,touch,idle}} \triangleq {}_{y}r^{i+1}_{\text{core,touch}} = - $$

- Terminal condition:
  ${}^{r}f^{\tau}_{\text{core,touch,idle}} \triangleq {}^{c}_{x}r^{i}_{\text{core,touch}}[\text{cmd}] = \text{GET}$
  When above mentioned command is obtained from the control subsystem the virtual receptor stops being idle and immediately commences with the commanded get behaviour.

3.4.3 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,touch,get}}$ of the virtual receptor $r_{\text{core,touch}}$

- Transition function:

$$ {}^{r,c}f_{\text{core,touch,get}} \triangleq \begin{cases} {}^{c}_{y}r^{i+1}_{\text{core,touch}} = - & \text{for } \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,touch}}[\text{value}]\right) \\ {}^{c}_{y}r^{i+1}_{\text{core,touch}}[\text{value}] = {}^{R}_{x}r^{i}_{\text{core,touch}}[\text{value}] & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,touch}}[\text{value}]\right) \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.

$$ {}^{r,R}f_{\text{core,touch,get}} \triangleq $$

$$ \begin{cases} \left.\begin{cases} {}^{R}_{y}r^{i+1}_{\text{core,touch}}[\text{cmd}] & = & \text{GET\_DATA} \\ {}^{R}_{y}r^{i+1}_{\text{core,touch}}[\text{data\_name}] & = & {}^{c}_{x}r^{i}_{\text{core,touch}}[\text{arg}[\text{data\_name}]] \end{cases}\right\} & \text{for } i = i_0 \\ {}^{R}_{y}r^{i+1}_{\text{core,touch}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{r,r}f_{\text{core,touch,get}} \triangleq {}^{r}r^{i+1}_{\text{core,touch}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,touch}}[\text{value}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,touch}}[\text{value}]\right) \end{cases} $$

The transition function returns to the control subsystem ${}^{c}_{y}r_{\text{core,touch}}$ the current value of a parameter ${}^{c}_{y}r^{i+1}_{\text{core,touch}}[\text{value}]$ named as data_name.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,touch,get}} \triangleq {}^{r}r^{i}_{\text{core,touch}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{r}r^{i}_{\text{core,touch}}[\text{tm}]$ is switched when the value of a desired parameter is obtained from the real receptor $R_{\text{core,touch}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.4.4 FSM governing the virtual receptor $r_{\text{core,touch}}$

The two state automaton (FSM) governing the activities of the touch virtual receptor $r_{\text{core,touch}}$ is presented in fig. 8.

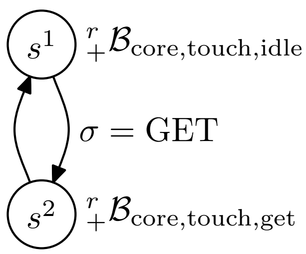

Figure 8: FSM governing the activities of the touch virtual receptor $r_{\text{core,touch}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}r^{i}_{\text{core,touch}}[\text{cmd}]$

3.5 Virtual receptor $r_{\text{core,inertial}}$

The virtual receptor $r_{\text{core,inertial}}$ gathers data fom the accelerometer and gyroscope. It may return to the control subsystem the current 3-axis revolute velocities and accelerations the center of the body with respect to the robot torso. Moreover, it provides the calculated orientation angles of the body using the data acquired from the gyroscope and the accelerometer.

3.5.1 Communication buffers and internal memory of the virtual receptor $r_{\text{core,inertial}}$

- Internal memory ${}^{r}r_{\text{core,inertial}}$:
  tm – termination marker

- Real receptor control ${}^{R}_{y}r_{\text{core,inertial}}$:
  cmd – command from the virtual receptor,
  data_name – data name of a parameter stored in a NAOqi ALMemory module,

- Input from the real receptor ${}^{R}_{x}r_{\text{core,inertial}}$:
  value – current value of a parameter named as ${}^{c}_{x}r_{\text{core,inertial}}[\text{data\_name}]$; this value is transmitted to the control subsystem in response to its query

- Input from the control subsystem ${}^{c}_{x}r_{\text{core,inertial}}$:
  cmd – command from the control subsystem; cmd $\in$ {GET},
  arg – arguments from obtained from the control subsystem;
  arg = [data_name], where:
  data_name – is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module

- Output produced by the virtual receptor for the control subsystem ${}^{c}_{y}r_{\text{core,inertial}}$:
  value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

3.5.2 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,inertial,idle}}$ of the virtual receptor $r_{\text{core,inertial}}$

- Transition function:

$$ {}^{r}f_{\text{core,inertial,idle}} \triangleq {}_{y}r^{i+1}_{\text{core,inertial}} = - $$

- Terminal condition:
  ${}^{r}f^{\tau}_{\text{core,touch,idle}} \triangleq {}^{c}_{x}r^{i}_{\text{core,inertial}}[\text{cmd}] = \text{GET}$
  When the above mentioned command is obtained from the control subsystem the virtual receptor stops being idle and immediately commences with the commanded send behaviour.

3.5.3 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,inertial,get}}$ of the virtual receptor $r_{\text{core,inertial}}$

- Transition function:

$$ {}^{r,c}f_{\text{core,inertial,get}} \triangleq \begin{cases} {}^{c}_{y}r^{i+1}_{\text{core,inertial}} = - & \text{for } \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,inertial}}[\text{value}]\right) \\ {}^{c}_{y}r^{i+1}_{\text{core,inertial}}[\text{value}] = {}^{R}_{x}r^{i}_{\text{core,inertial}}[\text{value}] & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,inertial}}[\text{value}]\right) \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.
${}^{r,R}f_{\text{core,inertial,get}} \triangleq$

$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{R}_{y}r^{i+1}_{\text{core,inertial}}[\text{cmd}] & = & \text{GET\_DATA} \\ {}^{R}_{y}r^{i+1}_{\text{core,inertial}}[\text{data\_name}] & = & {}^{c}_{x}r^{i}_{\text{core,inertial}}[\text{arg}[\text{data\_name}]] \end{array} \right\} & \text{for } i = i_0 \\ {}^{R}_{y}r^{i+1}_{\text{core,inertial}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{r,r}f_{\text{core,inertial,get}} \triangleq {}^{r}r^{i+1}_{\text{core,inertial}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,inertial}}[\text{value}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,inertial}}[\text{value}]\right) \end{cases} $$

The transition function returns to the control subsystem ${}^{c}_{y}r_{\text{core,inertial}}$ the current value of a parameter ${}^{c}_{y}r^{i+1}_{\text{core,inertial}}[\text{value}]$ named as data_name.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,inertial,get}} \triangleq {}^{r}r^{i}_{\text{core,inertial}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{r}r^{i}_{\text{core,inertial}}[\text{tm}]$ is switched when the value of a desired parameter is obtained from the real receptor $R_{\text{core,inertial}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.5.4 FSM governing the virtual receptor $r_{\text{core,inertial}}$

The two state automaton (FSM) governing the activities of the inertial virtual receptor $r_{\text{core,inertial}}$ is presented in fig. 9.

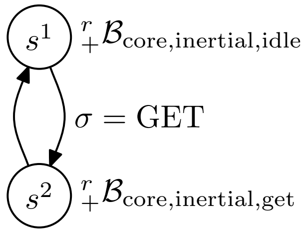

Figure 9: FSM governing the activities of the inertial virtual receptor $r_{\text{core,inertial}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}r^{i}_{\text{core,inertial}}[\text{cmd}]$

3.6 Virtual receptor $r_{\text{core,fsr}}$

The virtual receptor $r_{\text{core,fsr}}$ acquires data from the Force Sensitive Resistors. Each foot contains four such receptors. Those receptors measure the resistance change due to the pressure applied. The virtual receptor may transmit to the control subsystem the measurement of each sensor, the total weight supported by each leg and the location of the center of pressure of each leg.

3.6.1 Communication buffers and internal memory of the virtual receptor $r_{\text{core,fsr}}$

- Internal memory ${}^{r}r_{\text{core,fsr}}$:
  tm – termination marker
- Real receptor control ${}^{R}_{y}r_{\text{core,fsr}}$:
  cmd – command from the virtual receptor,
  data_name – data name of a parameter stored in a NAOqi ALMemory module,
- Input from the real receptor ${}^{R}_{x}r_{\text{core,fsr}}$:
  value – current value of a parameter named as ${}^{c}_{x}r_{\text{core,fsr}}[\text{data\_name}]$; this value is transmitted to the control subsystem in response to its query

- Input from the control subsystem ${}^{c}_{x}r_{\text{core,fsr}}$:
  cmd – command from the control subsystem; cmd $\in$ {GET},
  arg – arguments from the control subsystem;
  arg = [data_name], where:
  data_name – is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module
- Output produced by the virtual receptor for the control subsystem ${}^{c}_{y}r_{\text{core,fsr}}$:
  value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

3.6.2 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,fsr,idle}}$ of the virtual receptor $r_{\text{core,fsr}}$

- Transition function:

$$ {}^{r}f_{\text{core,fsr,idle}} \triangleq {}_{y}r^{i+1}_{\text{core,fsr}} = - $$

- Terminal condition:
  ${}^{r}f^{\tau}_{\text{core,fsr,idle}} \triangleq {}^{c}_{x}r^{i}_{\text{core,fsr}}[\text{cmd}] = \text{GET}$
  When above mentioned command is obtained from the control subsystem the virtual receptor stops being idle and immediately commences with the commanded send behaviour.

3.6.3 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,fsr,get}}$ of the virtual receptor $r_{\text{core,fsr}}$

- Transition function:

$$ {}^{r,c}f_{\text{core,fsr,get}} \triangleq \begin{cases} {}^{c}_{y}r^{i+1}_{\text{core,fsr}} = - & \text{for } \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,fsr}}[\text{value}]\right) \\ {}^{c}_{y}r^{i+1}_{\text{core,fsr}}[\text{value}] = {}^{R}_{x}r^{i}_{\text{core,fsr}}[\text{value}] & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,fsr}}[\text{value}]\right) \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.
${}^{r,R}f_{\text{core,fsr,get}} \triangleq$

$$ \begin{cases} \left.\begin{cases} {}^{R}_{y}r^{i+1}_{\text{core,fsr}}[\text{cmd}] & = \text{GET\_DATA} \\ {}^{R}_{y}r^{i+1}_{\text{core,fsr}}[\text{data\_name}] & = {}^{c}_{x}r^{i}_{\text{core,fsr}}[\text{arg}[\text{data\_name}]] \end{cases}\right\} & \text{for } i = i_0 \\ {}^{R}_{y}r^{i+1}_{\text{core,fsr}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{r,r}f_{\text{core,fsr,get}} \triangleq {}^{r}r^{i+1}_{\text{core,fsr}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,fsr}}[\text{value}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,fsr}}[\text{value}]\right) \end{cases} $$

The transition function returns to the control subsystem ${}^{c}_{y}r_{\text{core,fsr}}$ the current value of the parameter ${}^{c}_{y}r^{i+1}_{\text{core,fsr}}[\text{value}]$ named as data_name.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,fsr,get}} \triangleq {}^{r}r^{i}_{\text{core,fsr}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{r}r^{i}_{\text{core,fsr}}[\text{tm}]$ is switched when the value of the desired parameter is obtained from the real receptor $R_{\text{core,fsr}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.6.4 FSM governing the virtual receptor $r_{\text{core,fsr}}$

The two state automaton (FSM) governing the activities of the Force Sensitive Resistors virtual receptor $r_{\text{core,fsr}}$ is presented in fig. 10.

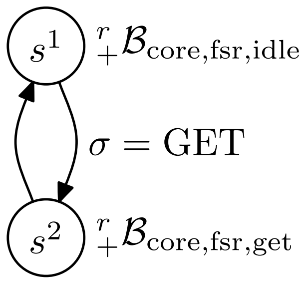

Figure 10: FSM governing the activities of the Force Sensitive Resistors virtual receptor $r_{\text{core,fsr}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}r^{i}_{\text{core,fsr}}[\text{cmd}]$

3.7 Virtual receptor $r_{\text{core,cam}}$

The virtual receptor $r_{\text{core,cam}}$ is responsible for acquiring data obtained by the camera.

3.7.1 Communication buffers and internal memory of the virtual receptor $r_{\text{core,cam}}$

- Internal memory ${}^{r}r_{\text{core,cam}}$:
  - tm – termination marker,
  - name_id – subscriber identifier

- Real receptor control ${}^{R}_{y}r_{\text{core,cam}}$:
  - cmd – command from the virtual receptor,
  - data_name – data name of a parameter stored in a NAOqi ALMemory module,
  - value – a new value of camera parameter,
  - params – camera parameters transmitted to the real receptor,
    $\text{params} = [\text{res}, \text{cid}, \text{pf}, \text{cs}]$, where:
    - res – resolution,
    - cid – camera id,
    - pf – picture format,
    - cs – color space,
    - fr – frame rate,

- Input from the real receptor ${}^{R}_{x}r_{\text{core,cam}}$:
  - value – current value of a parameter named as ${}^{c}_{x}r_{\text{core,fsr}}[\text{data\_name}]$; this value is transmitted to the control subsystem in response to its query
  - image – image collected by the camera,
  - name_id – subscriber identifier,

- Input from the control subsystem ${}^{c}_{x}r_{\text{core,cam}}$:

  cmd – command from the control subsystem;
  cmd $\in$ {GET, IMAGE, SET_PARAMETERS},

  arg – arguments from the control subsystem;
  arg = [params, data_name, value], where:

  - params – camera parameters;
    params = [res, cid], where:
    - res – resolution,
    - cid – camera id,
  - data_name – is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module, possible parameters: resolution, picture format, color space, frame rate,
  - value – a new value of camera parameter,

- Output produced by the virtual receptor for the control subsystem ${}^{c}_{y}r_{\text{core,cam}}$:

  value – current value of a NAOqi parameter, named as data_name, received from the AlMemory module.

  image – file containing an image,

3.7.2 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,cam,idle}}$ of the virtual receptor $r_{\text{core,cam}}$

- Transition function:

$$ {}^{r}f_{\text{core,cam,idle}} \triangleq {}_{y}r^{i+1}_{\text{core,cam}} = - $$

- Terminal condition:

$$ \begin{aligned} {}^{r}f^{\tau}_{\text{core,cam,idle}} \triangleq\ & ({}^{c}_{x}r^{i}_{\text{core,cam}}[\text{cmd}] = \text{GET}) \vee ({}^{c}_{x}r^{i}_{\text{core,cam}}[\text{cmd}] = \text{IMAGE}) \vee \\ & ({}^{c}_{x}r^{i}_{\text{core,cam}}[\text{cmd}] = \text{SET\_PARAMETERS}) \end{aligned} $$

When any of the above mentioned commands is obtained from the control subsystem the virtual receptor stops being idle and immediately transits to an adequate state.

3.7.3 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,cam,image}}$ of the virtual receptor $r_{\text{core,cam}}$

- Transition functions:

$$ {}^{r,R}f_{\text{core,cam,image}} \triangleq $$

$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{cmd}] & = & \text{SUBSCRIBE} \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{params[res]}] & = & {}^{c}_{x}r^{i}_{\text{core,cam}}[\text{arg[params[res]]}] \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{params[cid]}] & = & {}^{c}_{x}r^{i}_{\text{core,cam}}[\text{arg[params[cid]]}] \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{params[cs]}] & = & \text{kBGRColorSpace} \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{params[fr]}] & = & \text{maxCameraFPS} \end{array} \right\} & \text{for } i = i_0 \\ \left\{ \begin{array}{lcl} {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{cmd}] & = & \text{GET\_IMAGE} \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{name\_id}] & = & {}^{r}r^{i}_{\text{core,cam}}[\text{name\_id}] \end{array} \right\} & \begin{array}{l} \text{for } i \neq i_0 \wedge \\ \text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{name\_id}]\right) \end{array} \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{cmd}] = \text{UNSUBSCRIBE} & \begin{array}{l} \text{for } i \neq i_0 \wedge \\ \text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]\right) \end{array} \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.

$$
{}^{r,r}f_{\text{core,cam,image}} \triangleq \\
\begin{cases}
{}^{r}r^{i+1}_{\text{core,cam}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]\right) \end{cases} \\
{}^{r}r^{i+1}_{\text{core,cam}}[\text{name\_id}] = {}^{R}_{x}r^{i}_{\text{core,cam}}[\text{name\_id}] \quad \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{name\_id}]\right)
\end{cases}
$$

$$
{}^{r,c}f_{\text{core,cam,image}} \triangleq \begin{cases} {}^{c}_{y}r^{i+1}_{\text{core,cam}} = - & \text{for } \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]\right) \\ {}^{c}_{y}r^{i+1}_{\text{core,cam}}[\text{image}] = \text{img}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]\right) & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]\right) \end{cases}
$$

These transition functions transfer to the real receptor $R_{\text{core,cam}}$ the three types of commands: SUBSCRIBE, GET_IMAGE and UNSUBSCRIBE and supplement them with such parameters as: camera parameters and name_id identifying the subscriber. After subscription to the camera, the real receptor $R_{\text{core,cam}}$ returns the name_id. The real receptor takes a photo in a format acceptable by OpenCV, but the control subsystem c_{core} requires the format used by ROS. When the new data appears in the buffer ${}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]$ the virtual receptor converts this data using img function, sends it to the control subsystem and, moreover, sends to the real receptor $R_{\text{core,cam}}$ the UNSUBSCRIBE command.

- Terminal condition:

$$
{}^{r}f^{\tau}_{\text{core,cam,image}} \triangleq {}^{r}r^{i}_{\text{core,cam}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{r}r^{i}_{\text{core,cam}}[\text{tm}]$ is switched when a new value appears in the ${}^{R}_{x}r^{i}_{\text{core,cam}}[\text{image}]$ buffer. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.7.4 Behaviour ${}^{e}_{+}\mathcal{B}_{\text{core,cam,set\_params}}$ of the virtual receptor $r_{\text{core,cam}}$

- Transition function:

$$
{}^{r,R}f_{\text{core,cam,set\_params}} \triangleq \begin{cases} {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{cmd}] & = \text{SET\_PARAMETERS} \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{data\_name}] & = {}^{c}_{x}r^{i}_{\text{core,cam}}[\text{arg}[\text{data\_name}]] \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{value}] & = {}^{c}_{x}r^{i}_{\text{core,cam}}[\text{arg}[\text{value}]] \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{params}[\text{cid}]] & = {}^{c}_{x}r^{i}_{\text{core,cam}}[\text{arg}[\text{params}[\text{cid}]]] \end{cases}
$$

This transition function transfers to the real receptor $R_{\text{core,cam}}$ the parameters obtained from the control subsystem. The parameter data_name for a given camera ${}^{c}_{x}r^{i}_{\text{core,cam}}[\text{arg}[\text{params}[\text{cid}]]]$ is changed to the provided value. It should be noted that all the activities performed by the NAOqi setParameter and other functions are executed within the real receptor $R_{\text{core,cam}}$.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,ls,set\_params}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.7.5 Behaviour ${}^{r}_{+}\mathcal{B}_{\text{core,cam,get}}$ of the virtual receptor $r_{\text{core,cam}}$

- Transition function:

$$ {}^{r,c}f_{\text{core,cam,get}} \triangleq \begin{cases} {}^{c}_{y}r^{i+1}_{\text{core,cam}} = - & \text{for } \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{value}] \right) \\ {}^{c}_{y}r^{i+1}_{\text{core,cam}}[\text{value}] = {}^{R}_{x}r^{i}_{\text{core,cam}}[\text{value}] & \text{for new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{value}] \right) \end{cases} $$

where new is a predicate being TRUE when a new value of its argument is obtained.

$$ {}^{r,R}f_{\text{core,cam,get}} \triangleq $$

$$ \begin{cases} \left. \begin{cases} {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{cmd}] & = \text{GET\_DATA} \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}}[\text{data\_name}] & = {}^{c}_{x}r^{i}_{\text{core,fsr}}[\text{arg}[\text{data\_name}]] \end{cases} \right\} & \text{for } i = i_0 \\ {}^{R}_{y}r^{i+1}_{\text{core,cam}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{r,r}f_{\text{core,cam,get}} \triangleq {}^{r}r^{i+1}_{\text{core,cam}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{value}] \right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{R}_{x}r^{i}_{\text{core,cam}}[\text{value}] \right) \end{cases} $$

The transition function returns to the control subsystem ${}^{c}_{y}r_{\text{core,cam}}$ the current value of the parameter ${}^{c}_{y}r^{i+1}_{\text{core,cam}}[\text{value}]$ named as data_name.

- Terminal condition:

$$ {}^{r}f^{\tau}_{\text{core,cam,get}} \triangleq {}^{r}r^{i}_{\text{core,cam}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{r}r^{i}_{\text{core,cam}}[\text{tm}]$ is switched when the value of the desired parameter is obtained from the real receptor $R_{\text{core,cam}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.7.6 FSM governing the virtual receptor $r_{\text{core,cam}}$

The four state automaton (FSM) governing the activities of the camera virtual receptor $r_{\text{core,cam}}$ is presented in fig. 11.

3.8 Control subsystem $c_{\text{core,cs}}$

The control subsystem $c_{\text{core,cs}}$ is responsible for communication with its virtual effectors and virtual receptors as well as other agents such as the repository agent or the cloud agent. It downloads and initiates the dynamic agent. Moreover, it provides to the dynamic agent the interface to the robot effectors and receptors at the ontological level adequate to the necessities of the executed task.

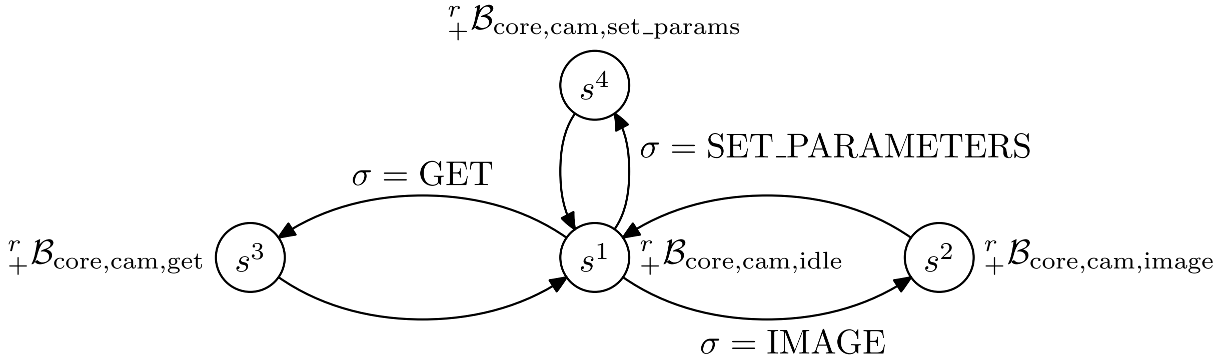

Figure 11: FSM governing the activities of the camera virtual receptor $r_{\text{core,cam}}$ of the core agent a_{core}; $\sigma \triangleq {}^{c}_{x}e^{i}_{\text{core,cam}}[\text{cmd}]$

3.8.1 Communication buffers and internal memory of the control subsystem $c_{\text{core,cs}}$

- Internal memory ${}^{c}c_{\text{core,cs}}$:

  tm – termination marker,
  threshold – threshold required for short command interpretation,
  app_name – application name to be downloaded from RAPP Store,
  app – list of pairs,
  app = [(word, app_name), ...], where:
  word – keyword for the application_name application,
  app_name – name of the application to be downloaded,
  language – language of the sound synthesis,
  dictionary – a vector of words that are to be recognized in the virtual receptor $r_{\text{core,mic}}$. The recognized_word will be used to download a desired dynamic agent (application),
  recog_word – recognized word in the virtual receptor $r_{\text{core,mic}}$,
  recog_sentence – the recognized sentence from the RAPP Platform,
  rd – contains the path to the recorded file,
  package – downloaded Dynamic Agent package from RAPP Platform,
  da_status – dynamic agent status,

- Output communication buffer ${}^{e}_{y}c_{\text{core,cs,ls}}$ controlling the virtual effector $e_{\text{core,ls}}$:

$$ {}^{e}_{y}c_{\text{core,cs,ls}} = {}^{c}_{x}e_{\text{core,ls}} $$

- Output communication buffer ${}^{e}_{y}c_{\text{core,cs,body}}$ controlling the virtual effector $e_{\text{core,body}}$:

$$ {}^{e}_{y}c_{\text{core,cs,body}} = {}^{c}_{x}e_{\text{core,body}} $$

- Input communication buffer ${}^{e}_{x}c_{\text{core,cs,ls}}$ obtaining proprioceptive information from the virtual effector $e_{\text{core,ls}}$:

$$ {}^{e}_{x}c_{\text{core,cs,ls}} = {}^{c}_{y}e_{\text{core,ls}} $$

- Input communication buffer ${}^{e}_{x}c_{\text{core,cs,body}}$ obtaining proprioceptive information from the virtual effector $e_{\text{core,body}}$:

$$ {}^{e}_{x}c_{\text{core,cs,body}} = {}^{c}_{y}e_{\text{core,body}} $$

- Output communication buffer ${}^{r}_{y}c_{\text{core,cs,mic}}$ controlling the virtual receptor $r_{\text{core,mic}}$:

$$ {}^{r}_{y}c_{\text{core,cs,mic}} = {}^{c}_{x}r_{\text{core,mic}} $$

- Output communication buffer ${}^{r}_{y}c_{\text{core,cs,inertial}}$ controlling the virtual receptor $r_{\text{core,inertial}}$:

$$ {}^{r}_{y}c_{\text{core,cs,inertial}} = {}^{c}_{x}r_{\text{core,inertial}} $$

- Output communication buffer ${}^{r}_{y}c_{\text{core,cs,touch}}$ controlling the virtual receptor $r_{\text{core,touch}}$:

$$ {}^{r}_{y}c_{\text{core,cs,touch}} = {}^{c}_{x}r_{\text{core,touch}} $$

- Output communication buffer ${}^{r}_{y}c_{\text{core,cs,fsr}}$ controlling the virtual receptor $r_{\text{core,fsr}}$:

$$ {}^{r}_{y}c_{\text{core,cs,fsr}} = {}^{c}_{x}r_{\text{core,fsr}} $$

- Output communication buffer ${}^{r}_{y}c_{\text{core,cs,cam}}$ controlling the virtual receptor $r_{\text{core,cam}}$:

$$ {}^{r}_{y}c_{\text{core,cs,cam}} = {}^{c}_{x}r_{\text{core,cam}} $$

- Input communication buffer ${}^{r}_{x}c_{\text{core,cs,mic}}$ obtaining aggregated information from the virtual receptor $r_{\text{core,mic}}$:

$$ {}^{r}_{x}c_{\text{core,cs,mic}} = {}^{c}_{y}r_{\text{core,mic}} $$

- Input communication buffer ${}^{r}_{x}c_{\text{core,cs,touch}}$ obtaining aggregated information from the virtual receptor $r_{\text{core,touch}}$:

$$ {}^{r}_{x}c_{\text{core,cs,touch}} = {}^{c}_{y}r_{\text{core,touch}} $$

- Input communication buffer ${}^{r}_{x}c_{\text{core,cs,inertial}}$ obtaining aggregated information from the virtual receptor $r_{\text{core,inertial}}$:

$$ {}^{r}_{x}c_{\text{core,cs,inertial}} = {}^{c}_{y}r_{\text{core,inertial}} $$

- Input communication buffer ${}^{r}_{x}c_{\text{core,cs,fsr}}$ obtaining aggregated information from the virtual receptor $r_{\text{core,fsr}}$:

$$ {}^{r}_{x}c_{\text{core,cs,fsr}} = {}^{c}_{y}r_{\text{core,fsr}} $$

- Input communication buffer ${}^{r}_{x}c_{\text{core,cs,cam}}$ obtaining aggregated information from the virtual receptor $r_{\text{core,cam}}$:

$$ {}^{r}_{x}c_{\text{core,cs,cam}} = {}^{c}_{y}r_{\text{core,cam}} $$

- Input from the dynamic agent agent ${}^{T}_{x}c_{\text{core,cs,da}}$:

da_status	–	status of dynamic agent,
cmd	–	command from dynamic agent ${}^{T}_{x}c_{\text{core,cs}}$ to control subsystem $c_{\text{core,cs}}$, $\text{cmd} \in \{\text{Terminate}, \text{Call}\}$
path	–	a motion trajectory for the moveAlongPath behaviour,
pose	–	current robot global pose with respect to world coordinate frame, $\text{pose} = [\text{position}, \text{orientation}]$, where:

position	–	robot position, $\text{position} = [\text{x}, \text{y}, \text{z}]$, where:
		x – x coordinate of a current position,
		y – y coordinate of a current position,
		z – z coordinate of a current position,
orientation	–	current robot orientation in a quaternion form,
		x – x component of a current robot orientation represented in a quaternion form,
		y – y component of a current robot orientation represented in a quaternion form,,
		z – z component of a current robot orientation represented in a quaternion form,,
		w – w component of a current robot orientation represented in a quaternion form,,

- arg_e_{ls} – arguments for a loudspeaker from a dynamic agent;
  $\text{arg\_e}_{\text{ls}} = [\text{cmd}, \text{arg}]$, where:
  - cmd – command for $e_{\text{core,ls}}$,
    $\text{cmd} \in \{\text{PLAY\_AUDIO}, \text{PLAY\_AUDIO}, \text{STOP\_SOUND}\}$,
  - arg – arguments for a virtual effector $e_{\text{core,ls}}$,
    $\text{arg} = [\text{text}, \text{fp}, \text{params}, \text{playLoop},$
    $\text{begin\_position}]$, where:
    - text – the text to be transformed into the synthesized sound,
    - fp – the path to the file that will be reproduced,
    - params – loudspeaker virtual effector parameters,
      $\text{params} = [\text{dvt}, \text{dl}, \text{dv}, \text{spr}]$,
      where:
      - dvt – desired voice type,
      - dl – desired language,
      - dv – volume requested from the range [0.0 - 1.0],
      - spr – stereo panorama requested (-1.0 : left, 1.0 : right, 0.0 : center),
    - playLoop – plays a file in a loop if the flag is set
      to TRUE, otherwise plays once,
    - begin_position – position in second where the playing
      should begin,

- arg_e_{body} – arguments for a body effector from a dynamic agent, $arg\_e_{body} = [cmd, arg]$, where:
  - cmd – command for body effector received from dynamic agent, $cmd \in \{MOVE\_TO, MOVE\_VEL, MOVE\_HEAD, TAKE\_PREDEFINED\_POSTURE, MOVE\_STOP, MOVE\_JOINT, LOOK\_AT\_POINT\}$,
  - arg – arguments from the control subsystem; $arg = [velocity, dpose, posture, dja, look\_at, move\_head]$, where:
    - velocity – velocity of motion with respect to the robot coordinate frame (memorized argument of the MOVE command); $velocity = [v_x, v_y, \omega]$, where:
      - v_x – velocity along the X-axis, in meters per second,
      - v_y – velocity along the Y-axis, in meters per second,
      - ω – velocity around the Z-axis, in radians per second,
    - dpose – desired position with respect to the robot coordinate frame (memorized argument of the MOVE command); $dpose = [x, y, \theta]$, where:
      - x – distance along the X-axis, in meters,
      - y – distance along the Y-axis, in meters,
      - θ – rotation around the Z-axis, in radians,
    - posture – name of a predefined posture to be attained,
    - dja – desired joint angles with parameters, $dja = [joints, values]$, where:
      - joints – a name or names of joints,
      - values – one or more angles in radians, during the interpolation of angles,
    - look_at – point at which the head of the robot should be directed, in a FRAME_WORLD coordinates, $look\_at = [x, y, z]$, where:
      - x – x coordinate of a desired point,
      - y – y coordinate of a desired point,
      - z – z coordinate of a desired point,
    - move_head – two desired angles needed to rotate the robot head, $move\_head = [yaw, pitch]$, where:
      - yaw – head end position in yaw angles,
      - pitch – head end position in pitch angles,

$\text{arg\_r}_{\text{mic}}$ – arguments for a microphone from a dynamic agent, $\text{arg\_r}_{\text{mic}} = [\text{cmd}, \text{arg}]$, where:

- cmd – command for a microphone receptor, $\text{cmd} \in \{\text{CAPTURE\_AUDIO}, \text{WORD\_SPOTTING}\}$,
- arg – arguments from control subsystem, $\text{arg} = [\text{dct}, \text{dict\_size}, \text{fp}, \text{time}, \text{enrg}, \text{st}]$, where:
  - dct – contains the list of words that should be recognized,
  - dict_size – dictionary size,
  - fp – file containing the recorded signal samples,
  - time – commanded duration of the recording,
  - enrg – threshold of signal energy for microphones,
  - st – time during which the microphone signal energy is compared with the threshold. When during this time the signal will be lower than the threshold then the recording stops,

$\text{arg\_r}_{\text{cam}}$ – arguments for a camera from a dynamic agent, $\text{arg\_r}_{\text{cam}} = [\text{cmd}, \text{arg}]$, where:

- cmd – command for a camera receptor, $\text{cmd} \in \{\text{CAPTURE\_IMAGE}, \text{SET\_CAMERA\_PARAMS}\}$,
- arg – arguments from the control subsystem; $\text{arg} = [\text{params}, \text{data\_name}, \text{value}]$, where:
  - params – camera parameters; $\text{params} = [\text{res}, \text{cid}]$, where:
    - res – resolution,
    - cid – camera id,
  - data_name – is a string that contains a name of a parameter stored in a NAOqi ALMemory module. It is used to query the value of the NAOqi parameter stored in the AlMemory module, possible parameters: resolution, picture format, color space, frame rate,
  - value – a new value of camera parameter,

- Input from the RAPP Platform agent ${}_{x}^{T}c_{\text{core,cs,rp}}$:

  recog_sentence – the recognized sentence from the RAPP Platform,

  downloaded – a boolean information if the package was downloaded correctly from RAPP Store. If a value is False then probably a package doesn't exist in the RAPP Store or a package was downloaded incorrectly,

- Output to the dynamic agent ${}_{x}^{T}c_{\text{core,cs,da}}$:

is_finished – TRUE if a current behaviour was finished,
ca_status – core agent status,
captured_image – captured image from desired camera,
path_to_audio – path to the recorded audio file,
recognized_word – recognized word,
pose – current robot global pose with respect to world coordinate frame, pose = [position, orientation], where:
  position – robot position, position = [x, y, z], where:
    x – x coordinate of a current position,
    y – y coordinate of a current position,
    z – z coordinate of a current position,
  orientation – current robot orientation in a quaternion form,
    x – x component of a current robot orientation represented in a quaternion form,
    y – y component of a current robot orientation represented in a quaternion form,,
    z – z component of a current robot orientation represented in a quaternion form,,
    w – w component of a current robot orientation represented in a quaternion form,,

- Output to the RAPP Platform agent ${}^{T}_{y}c_{\text{core,cs,rp}}$:
  package – a dynamic package to be downloaded from RAPP Store,
  path_to_audio – path to the recorded audio file,
  rd – contains raw data from the microphones,

3.8.2 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,init}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$^{c}f_{\text{core,cs,init}} \triangleq -$$

Agent creation and its initialization are external to the agent and as such are not represented within this model. It is assumed that from the point of a particular agent its creation is caused by an external entity and the process itself is not governed by a particular transition function. The implementation of this process brings about the creation of a hop service, ros services and rosbridge. The parameters stored in the configure files are loaded into the internal memory, e.g. dictionary ${}^{c}c_{\text{core,cs}}$[dictionary], the threshold ${}^{c}c_{\text{core,cs}}$[threshold] and the path ${}^{c}c_{\text{core,cs}}$[rd] where recorded file will be placed. For that purpose the function getFromConfigureFile() is used. All this is represented by a behaviour having an empty transition function as an argument. It is represented in the graph of the FSM governing the actions of the agent just for the sake of completeness, from the point of view of its implementation.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,init}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.8.3 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,register}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c}f_{\text{core,cs,register}} \triangleq - $$

This is a behaviour registering with the RAPP platform – currently not used.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,register}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.8.4 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,listen}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c,r}f_{\text{core,cs,listen}} \triangleq $$
$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{cmd}] & = & \text{RECOGNIZE} \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg}[\text{dictionary}]] & = & {}^{c}c^{i}_{\text{core,cs}}[\text{dictionary}] \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg}[\text{threshold}]] & = & {}^{c}c^{i}_{\text{core,cs}}[\text{threshold}] \end{array} \right\} & \text{for } i = i_0 \\ {}^{r}_{y}c^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0 \end{cases} $$

$$ {}^{c,c}f_{\text{core,cs,listen}} \triangleq $$
$$ \begin{cases} {}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right) \end{cases} \\ {}^{c}c^{i+1}_{\text{core,cs}}[\text{recog\_word}] = {}^{r}_{x}\text{c}^{\text{i}}_{\text{core,cs,mic}}[\text{rw}] \quad \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right) \end{cases} $$

It sets the dictionary and commands the microphones to listen to the user.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,listen}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the new value of a recognized word is obtained from the virtual receptor $r_{\text{core,mic}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.5 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,interpret}_{\text{Long}}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$
{}^{c,T}f_{\text{core,cs,interpret}_{\text{Long}}} \triangleq \begin{cases} {}^{T}_{y}c^{i+1}_{\text{core,cs,rp}}[\text{rd}] = {}^{c}c^{i}_{\text{core,cs}}[\text{rd}] & \text{for } i = i_0 \\ {}^{T}_{y}c^{i+1}_{\text{core,cs,rp}} = - & \text{for } i \neq i_0 \end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,interpret}_{\text{Long}}} \triangleq
$$

$$
\begin{cases} {}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{recog\_word}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{recog\_word}]\right) \end{cases} \\ {}^{c}c^{i+1}_{\text{core,cs}}[\text{recog\_sentence}] = \\ {}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{recog\_sentence}] \quad \text{for } i \neq i_0 \wedge \text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{recog\_sentence}]\right) \\ {}^{c}c^{i+1}_{\text{core,cs}}[\text{app\_name}] = \\ \text{getAppName}\left({}^{T}_{x}\text{c}^{\text{i}}_{\text{core,cs,rp}}[\text{recog\_sentence}]\right) \quad \text{for } i \neq i_0 \wedge \\ \qquad \text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{recog\_sentence}]\right) \end{cases}
$$

Sends to the RAPP Platform the recorded raw data to recognize a user command and interprets the recognized word. Function getAppName returns the application name ${}^{c}c^{i}_{\text{core,cs}}[\text{app}[\text{app\_name}]]$ based on a recognized word ${}^{c}c^{i}_{\text{core,cs}}[\text{recog\_word}]$.

- Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,interpret}_{\text{Long}}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the new value of a recognized word is obtained from the RAPP Platform. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.6 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,interpret}_{\text{short}}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$
{}^{c,c}f_{\text{core,cs,interpret}_{\text{short}}} \triangleq {}^{c}c^{i+1}_{\text{core,cs}}[\text{app\_name}] =
$$

$$
\begin{cases} \text{getAppName}\left({}^{c}\text{c}^{\text{i+1}}_{\text{core,cs}}[\text{recog\_word}]\right) & \text{for } {}^{c}c^{i+1}_{\text{core,cs}}[\text{recog\_word}] \neq \text{Abort} \wedge \\ & {}^{c}c^{i+1}_{\text{core,cs}}[\text{recog\_word}] \neq \text{Empty} \wedge \\ & {}^{c}c^{i+1}_{\text{core,cs}}[\text{recog\_word}] \neq \text{Long} \\ \text{Error} & \text{for otherwise} \end{cases}
$$

Interprets the short command. Function getAppName returns the application name ${}^{c}c^{i}_{\text{core,cs}}[\text{app}[\text{app\_name}]]$ based on a recognized word ${}^{c}c^{i}_{\text{core,cs}}[\text{recog\_word}]$.

- Terminal condition:
${}^{c}f^{\tau}_{\text{core,cs,interpret}_{\text{short}}} \triangleq \text{TRUE}$
This is a one step behaviour, so the terminal condition is TRUE.

3.8.7 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,inform}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

- Transition function:

$$
{}^{c,e}f_{\text{core,cs,inform}} \triangleq
$$

$$
\begin{cases}
\left\{\begin{array}{lcl}
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{cmd}] & = & \text{SAY} \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[text]}] & = & \text{"Command\_was\_not\_recognized"}
\end{array}\right\} & \text{for } i = i_0 \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,inform}} \triangleq
$$

$$
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] =
\begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,ls}}[\text{reply}]\right) \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,ls}}[\text{reply}]\right)
\end{cases}
$$

Behaviour calls a service from the virtual effector to inform that the command was not recognized.

- Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,inform}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the sound was synthesized. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.8 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,load}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$
{}^{c,T}f_{\text{core,cs,load}} \triangleq
\begin{cases}
{}^{T}_{y}c^{i+1}_{\text{core,cs}}[\text{package}] = & \\
{}^{c}c^{i}_{\text{core,cs,rp}}[\text{app\_name}] & \text{for } i = i_0 \\
{}^{T}_{y}c^{i+1}_{\text{core,cs,rp}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,load}} \triangleq
$$

$$
\begin{cases}
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] =
\begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{downloaded}]\right) \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{downloaded}]\right)
\end{cases} \\
{}^{c}c^{i+1}_{\text{core,cs}}[\text{package}] = \text{downloadDAPackage()} \quad
\begin{array}{l}
\text{for } i \neq i_0 \wedge \text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{downloaded}]\right) \\
\wedge\, {}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{downloaded}] = \text{TRUE}
\end{array}
\end{cases}
$$

The recognized application name is sent as a request of downloading a package of dynamic agent. Behaviour downloads also a dynamic agent package from the RAPP Platform.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,load}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the information is received if the package was downloaded correctly from the RAPP Store. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.9 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,activate}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c}f_{\text{core,cs,activate}} \triangleq \text{activateDA}({}^{c}\text{c}^{\text{i+1}}_{\text{core,cs}}[\text{package}]) $$

$$ {}^{c,c}f_{\text{core,cs,activate}} \triangleq $$
$$ \begin{cases} {}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{T}_{x}c^{i+1}_{\text{core,cs,da}}[\text{da\_status}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{T}_{x}c^{i+1}_{\text{core,cs,da}}[\text{da\_status}]\right) \end{cases} \\ {}^{c}c^{i+1}_{\text{core,cs}}[\text{da\_status}] = {}^{T}_{x}\text{c}^{\text{i+1}}_{\text{core,cs,da}}[\text{da\_status}] \quad \text{for } i \neq i_0 \wedge \\ \qquad \text{new}\left({}^{T}_{x}c^{i+1}_{\text{core,cs,da}}[\text{da\_status}]\right) \\ \qquad {}^{T}_{x}c^{i+1}_{\text{core,cs,da}}[\text{da\_status}] = \text{Working} \end{cases} $$

Activates the dynamic agent process.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,activate}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the information is received from the dynamic agent that it was activated and is ready to send commands. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.10 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,wait\_cmd}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c,e}f_{\text{core,cs,wait\_cmd}} \triangleq $$
$$ \begin{cases} {}^{c,e}f_{\text{core,cs,wait\_body}} \triangleq \\ \left.\begin{array}{lcl} {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{cmd}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{cmd}]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}]] \end{array}\right\} \text{for new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{cmd}]]\right) \\ {}^{c,e}f_{\text{core,cs,wait\_ls}} \triangleq \\ \left.\begin{array}{lcl} {}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{cmd}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{cmd}]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg}]] \end{array}\right\} \text{for new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{cmd}]]\right) \end{cases} $$

$$
{}^{c,r}f_{\text{core,cs,wait\_cmd}} \triangleq
\begin{cases}
{}^{c,r}f_{\text{core,cs,wait\_mic}} \triangleq \\
\left.\begin{array}{lcl}
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{cmd}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{cmd}]] \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg}]]
\end{array}\right\} \text{ for new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{cmd}]]\right) \\
{}^{c,r}f_{\text{core,cs,wait\_cam}} \triangleq \\
\left.\begin{array}{lcl}
{}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{cmd}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{cmd}]] \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{arg}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{arg}]]
\end{array}\right\} \text{ for new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{cmd}]]\right)
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,wait}_{\text{cmd}}} \triangleq
\begin{cases}
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] =
\begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{cmd}]\right) \\
 & \wedge\neg\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{cmd}]\right) \\
 & \wedge\neg\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{cmd}]]\right) \\
 & \wedge\neg\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{cmd}]]\right) \\
 & \wedge\, {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{cmd}] \neq \text{Terminate} \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\Big(\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{cmd}]\right) \\
 & \vee\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{cmd}]\right) \\
 & \vee\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{cmd}]]\right) \\
 & \vee\text{new}\left({}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{cmd}]]\right) \\
 & \vee\, {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{cmd}] = \text{Terminate}\Big)
\end{cases}
\end{cases}
$$

Behaviour waits for new commands and arguments from dynamic agent.

- Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,wait}_{\text{cmd}}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the new dynamic agent command appears in the control subsystem. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.11 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,execute}}$ of the control subsystem $c_{\text{core,cs}}$

Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,execute}}$ represents many behaviours called by dynamic agent. Below a list of them is presented:

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,textToSpeech}}$

Transition function:

$$
{}^{c,e}f_{\text{core,cs,textToSpeech}} \triangleq
$$

$$
\begin{cases}
\left\{\begin{array}{lcl}
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{cmd}] & = & \text{SAY} \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[text]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[text]}]] \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[params[dl]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[params[dl]]}]]
\end{array}\right\} & \text{for } i = i_0 \\
{}^{e}_{y}c^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,textToSpeech}} \triangleq
$$

$$
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,ls}}[\text{synthesized}]\right) \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,ls}}[\text{synthesized}]\right)
\end{cases}
$$

$$
{}^{c,T}f_{\text{core,cs,textToSpeech}} \triangleq {}^{T}_{y}c^{i+1}_{\text{core,cs,da}}[\text{is\_finished}] = \begin{cases}
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,ls}}[\text{synthesized}]\right) \\
- & \text{otherwise}
\end{cases}
$$

The command SAY and parameters are transfered to the virtual effector $e_{\text{core,ls}}$.

Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,textToSpeech}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched if the sound was synthesized. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,playAudio}}$

Transition function:

$$
{}^{c,e}f_{\text{core,cs,playAudio}} \triangleq
$$

$$
\left\{\begin{array}{lcl}
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{cmd}] & = & \text{PLAY} \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[fp]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[fp]}]] \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[playLoop]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[playLoop]}]] \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[begin\_position]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[begin\_position]}]] \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[params[dv]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[params[dv]]}]] \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,ls}}[\text{arg[params[spr]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{ls}}[\text{arg[params[spr]]}]]
\end{array}\right.
$$

$$
{}^{c,c}f_{\text{core,cs,textToSpeech}} \triangleq {}^{c}c^{i+1}_{\text{core,cs}}[\text{next\_state}] = \text{WAIT\_CMD}
$$

The command PLAY and parameters are transfered to the virtual effector $e_{\text{core,ls}}$.

Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,playAudio}} \triangleq \text{TRUE}
$$

This is a one step behaviour, so the terminal condition is TRUE.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,wordSpotting}}$
  Transition function:

$$
{}^{c,r}f_{\text{core,cs,wordSpotting}} \triangleq
\begin{cases}
\left.\begin{cases}
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{cmd}] & = & \text{RECOGNIZE} \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[dictionary]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg[dct]}]]
\end{cases}\right\} & \text{for } i = i_0 \\
{}^{r}_{y}c^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,wordSpotting}} \triangleq
\begin{cases}
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right) \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right)
\end{cases} \\
{}^{c}c^{i+1}_{\text{core,cs}}[\text{recog\_word}] = \\
{}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}] \quad \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right)
\end{cases}
$$

$$
{}^{c,T}f_{\text{core,cs,wordSpotting}} \triangleq {}^{T}_{y}c^{i+1}_{\text{core,cs,da}}[\text{recognized\_word}] =
\begin{cases}
{}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}] & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]\right) \\
- & \text{otherwise}
\end{cases}
$$

  It recognizes the word included in the dictionary and returns it to the dynamic agent.

  Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,wordSpotting}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

  The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the new value of a recognized word is obtained from the virtual receptor $r_{\text{core,mic}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,captureAudio}}$
  Transition function:

$$
{}^{c,r}f_{\text{core,cs,captureAudio}} \triangleq
\begin{cases}
\left.\begin{array}{lcl}
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{cmd}] & = & \text{RECORD} \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[fp]}] & = & {}^{c}c^{i}_{\text{core,cs}}[\text{rd}] \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[time]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg[time]}]]
\end{array}\right\} & \text{for } i = i_0 \wedge \text{new}\Big(\\
& {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg[time]}]]\Big) \\
\left.\begin{array}{lcl}
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{cmd}] & = & \text{REGISTER} \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[fp]}] & = & {}^{c}c^{i}_{\text{core,cs}}[\text{rd}] \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[st]}] & = & \\
{}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg[st]}]] & & \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[enrg]}] & = & \\
{}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg[enrg]}]] & & \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[params]}] & = & [\text{smpl\_rate, channels}]
\end{array}\right\} & \text{for } i = i_0 \wedge \neg\text{new}\Big(\\
& {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{mic}}[\text{arg[time]}]]\Big) \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,mic}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,captureAudio}} \triangleq
\begin{cases}
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rec}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rec}]\right) \end{cases} \\
{}^{c}c^{i+1}_{\text{core,cs}}[\text{next\_state}] = \\
\text{WAIT\_CMD} \qquad \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rec}]\right)
\end{cases}
$$

$$
{}^{c,T}f_{\text{core,cs,captureAudio}} \triangleq {}^{T}_{y}c^{i+1}_{\text{core,cs,da}}[\text{path\_to\_audio}] =
\begin{cases} {}^{c}c^{i}_{\text{core,cs}}[\text{rd}] & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rec}]\right) \\ - & \text{otherwise} \end{cases}
$$

It records the sound. RECORD command records sound for a given time, whereas REGISTER command is responsible for recording sound until the silence detection.

Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,captureAudio}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the new recording is terminated. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,getTransform}}$
  Returns the matrix as a transformation between two coordinates.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,captureImage}}$
  Transition function:

$$
{}^{c,r}f_{\text{core,cs,captureImage}} \triangleq
\begin{cases}
\left.\begin{cases}
{}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{cmd}] = \text{IMAGE} \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{arg[params[cid]]}] = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{arg[params[cid]]}]] \\
{}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{arg[params[res]]}] = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{arg[params[res]]}]]
\end{cases}\right\} & \text{for } i = i_0 \\
{}^{r}_{y}c^{i+1}_{\text{core,cam}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,captureImage}} \triangleq
\begin{cases}
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,cam}}[\text{image}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,cam}}[\text{image}]\right) \end{cases}
\end{cases}
$$

$$
{}^{c,T}f_{\text{core,cs,captureImage}} \triangleq {}^{T}_{y}c^{i+1}_{\text{core,cs,da}}[\text{captured\_image}] =
\begin{cases} {}^{r}_{x}c^{i}_{\text{core,cs,cam}}[\text{image}] & \text{for } i \neq i_0 \wedge \text{new}\left({}^{r}_{x}c^{i}_{\text{core,cs,cam}}[\text{image}]\right) \\ - & \text{otherwise} \end{cases}
$$

Captures the image from the robots camera and transfers the captured image to the dynamic agent.

Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,captureImage}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the new image is received from the virtual receptor $r_{\text{core,cam}}$. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,setCameraParams}}$
Transition function:

$$ {}^{c,r}f_{\text{core,cs,setCameraParams}} \triangleq $$
$$ \left\{ \begin{array}{lcl} {}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{cmd}] & = & \text{SET\_PARAMETERS} \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{arg[value]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{arg[value]}]] \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{arg[dataname]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{arg[data\_name]}]] \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,cam}}[\text{arg[params[cid]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_r}_{\text{cam}}[\text{arg[params[cid]]}]] \end{array} \right. \quad \text{for } i = i_0 $$

Modifies camera parameters.

Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,setCameraParams}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveTo}}$

Transition function:

$$ {}^{c,e}f_{\text{core,cs,moveTo}} \triangleq $$
$$ \left\{ \begin{array}{ll} \left\{ \begin{array}{lcl} {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{cmd}] & = & \text{MOVE\_TO} \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg[dpose[x]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg[dpose[x]]}]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg[dpose[y]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg[dpose[y]]}]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg[dpose[}\theta\text{]]}] & = & {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg[dpose[}\theta\text{]]}]] \end{array} \right\} & \text{for } i = i_0 \\ {}^{e}_{y}c^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0 \end{array} \right. $$

$$ {}^{c,c}f_{\text{core,cs,moveTo}} \triangleq $$
$$ \left\{ {}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \left\{ \begin{array}{ll} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,body}}[\text{pose}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,body}}[\text{pose}]\right) \end{array} \right. \right. $$

Move to the specified position with respect to the robot coordinate frame.

Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,moveTo}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE} $$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched if the robot reached the desired position. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveVel}}$
  Transition function:

$$
{}^{c,e}f_{\text{core,cs,moveVel}} \triangleq \begin{cases} {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{cmd}] & = \text{MOVE} \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}[\text{velocity}[\text{v}_\text{x}]]] & = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}[\text{velocity}[\text{v}_\text{x}]]]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}[\text{velocity}[\text{v}_\text{y}]]] & = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}[\text{velocity}[\text{v}_\text{y}]]]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}[\text{velocity}[\omega]]] & = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}[\text{velocity}[\omega]]]] \end{cases}
$$

  Move with specified velocity.

  Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,moveVel}} \triangleq \text{TRUE}
$$

  This is a one step behaviour, so the terminal condition is TRUE.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveStop}}$

  Transition function:

$$
{}^{c,e}f_{\text{core,cs,moveStop}} \triangleq {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{cmd}] = \text{STOP}
$$

  Robot stops movement.

  Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,moveStop}} \triangleq \text{TRUE}
$$

  This is a one step behaviour, so the terminal condition is TRUE.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveJoint}}$

  Transition function:

$$
{}^{c,e}f_{\text{core,cs,moveJoint}} \triangleq \begin{cases} \left.\begin{cases} {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{cmd}] & = \text{INTERPOLATION} \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}[\text{dja}[\text{joints}]]] & = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}[\text{dja}[\text{joints}]]]] \\ {}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}[\text{dja}[\text{values}]]] & = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}[\text{dja}[\text{values}]]]] \end{cases}\right\} & \text{for } i = i_0 \\ {}^{e}_{y}c^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0 \end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,moveJoint}} \triangleq
$$

$$
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,body}}[\text{attained}]\right) \\ \text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,body}}[\text{attained}]\right) \end{cases}
$$

  Move Nao joint to specified angle.

  Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,moveJoint}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched if the robot reached the desired angle position. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,takePredefinedPosture}}$

  Transition function:

$$
{}^{c,e}f_{\text{core,cs,takePredefinedPosture}} \triangleq
\begin{cases}
\left.\begin{cases}
{}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{cmd}] & = \text{POSTURE} \\
{}^{e}_{y}c^{i+1}_{\text{core,cs,body}}[\text{arg}[\text{posture}]] & = {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{arg\_e}_{\text{body}}[\text{arg}[\text{posture}]]]
\end{cases}\right\} & \text{for } i = i_0 \\
{}^{e}_{y}c^{i+1}_{\text{core,body}} = - & \text{for } i \neq i_0
\end{cases}
$$

$$
{}^{c,c}f_{\text{core,cs,takePredefinedPosture}} \triangleq
$$

$$
{}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] =
\begin{cases}
\text{FALSE} & \text{for } i = i_0 \\
- & \text{for } i \neq i_0 \wedge \neg\text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,body}}[\text{attained}]\right) \\
\text{TRUE} & \text{for } i \neq i_0 \wedge \text{new}\left({}^{e}_{x}c^{i}_{\text{core,cs,body}}[\text{attained}]\right)
\end{cases}
$$

  Move to a predefined posture.

  Terminal condition:

$$
{}^{c}f^{\tau}_{\text{core,cs,takePredefinedPosture}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}
$$

  The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched if the robot reached the desired posture. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,rest}}$
  Moves to a predefined safety posture and removes the joints stiffness.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveAlongPath}}$
  Robot moves along specified path.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,setGlobalPose}}$
  Sets a current robot position in a world frame.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,lookAtPoint}}$
  Robot looks at the point specified in world frame.

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,getRobotPose}}$
  Returns the current robot position.

3.8.12 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,destroy}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$
{}^{c}f_{\text{core,cs,destroy}} \triangleq \text{destroyDA}()
$$

  Kills all dynamic agent processes.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,destroy}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.8.13 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,unregister}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c}f_{\text{core,cs,unregister}} \triangleq \text{unregister}() $$

Unregisters robot from the repository agent.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,unregister}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.8.14 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,finish}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c}f_{\text{core,cs,finish}} \triangleq - $$

Kills all core agent processes. The destruction of the agent itself is beyond the agent model – it has to be caused by an outside source and does not require any internal behaviour, thus such a behaviour is represented by an empty transition function. It is represented in the graph of the FSM governing the actions of the agent just for the sake of completeness, from the point of view of its implementation.

- Terminal condition:

$$ {}^{c}f^{\tau}_{\text{core,cs,finish}} \triangleq \text{TRUE} $$

This is a one step behaviour, so the terminal condition is TRUE.

3.8.15 Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,record}_{\text{cmd}}}$ of the control subsystem $c_{\text{core,cs}}$

- Transition function:

$$ {}^{c,r}f_{\text{core,cs,record}_{\text{cmd}}} \triangleq $$
$$ \begin{cases} \left\{ \begin{array}{lcl} {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{cmd}] & = & \text{REGISTER} \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[file\_path]}] & = & {}^{c}c^{i}_{\text{core,cs}}[\text{rd}] \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[energy]}] & = & \text{ENERGY} \\ {}^{r}_{y}c^{i+1}_{\text{core,cs,mic}}[\text{arg[silence\_time]}] & = & \text{SILENCE\_TIME} \end{array} \right\} & \text{for } i = i_0 \\ {}^{r}_{y}c^{i+1}_{\text{core,mic}} = - & \text{for } i \neq i_0 \end{cases} $$

$${}^{c,c}f_{\text{core,cs,record}_{\text{cmd}}} \triangleq$$

$${}^{c}c^{i+1}_{\text{core,cs}}[\text{tm}] = \begin{cases} \text{FALSE} & \text{for } i = i_0 \\ - & \text{for } i \neq i_0 \land {}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rec}] \neq \text{TRUE} \\ \text{TRUE} & \text{for } i \neq i_0 \land {}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rec}] = \text{TRUE} \end{cases}$$

Behaviour calls a service from virtual receptor of recording until a silence detected.

- Terminal condition:

$${}^{c}f^{\tau}_{\text{core,cs,record}_{\text{cmd}}} \triangleq {}^{c}c^{i}_{\text{core,cs}}[\text{tm}] = \text{TRUE}$$

The termination marker ${}^{c}c^{i}_{\text{core,cs}}[\text{tm}]$ is switched when the virtual receptor $r_{\text{core,mic}}$ detects the silence. The value TRUE of the termination marker causes the termination of the behaviour in the next control step.

3.8.16 FSM governing the control subsystem $c_{\text{core,cs}}$

The fourteen state automaton (FSM) governing the activities of the body virtual effector $e_{\text{core,body}}$ is presented in fig. 12.

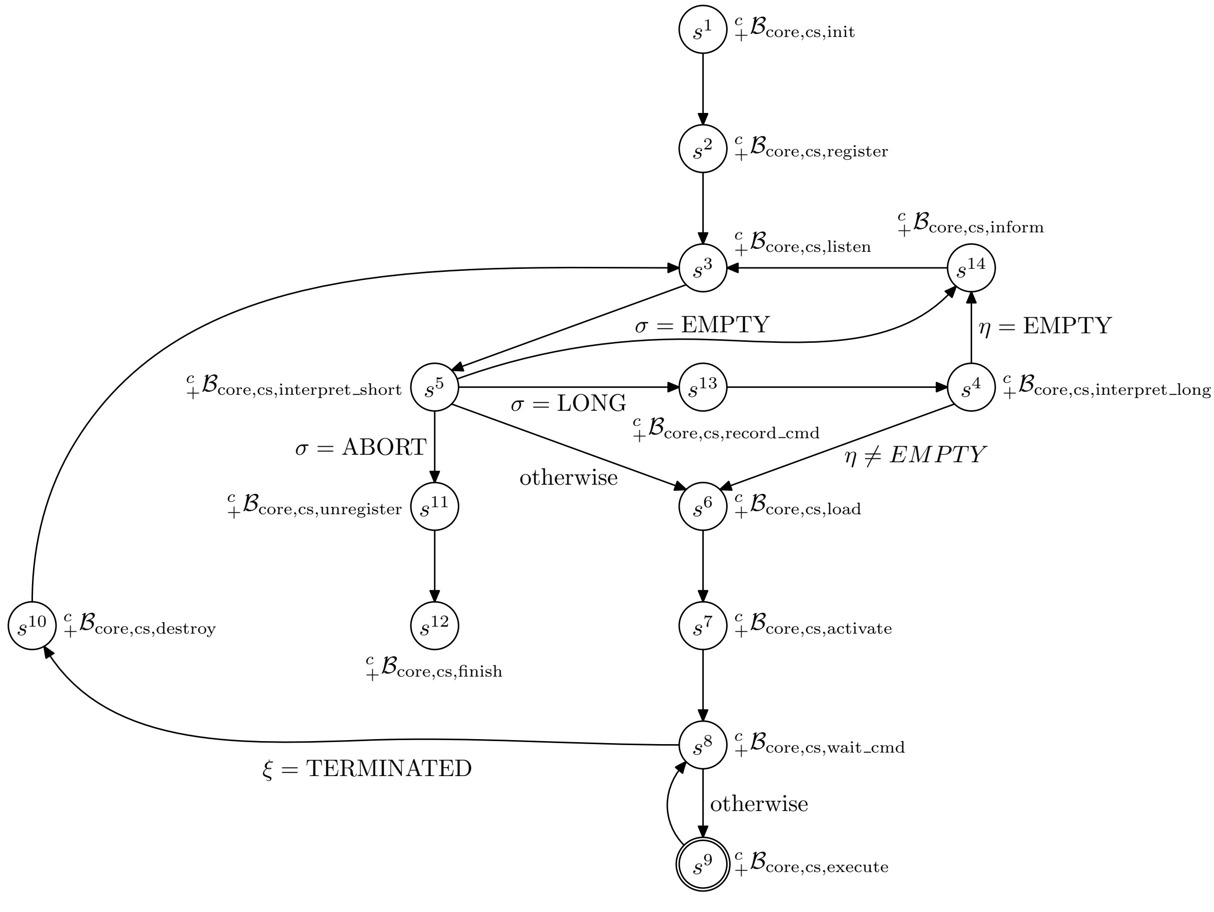

Figure 12: FSM governing the activities of the control subsystem c_{core} of the core agent a_{core}; $\sigma \triangleq {}^{r}_{x}c^{i}_{\text{core,cs,mic}}[\text{rw}]$, $\eta \triangleq {}^{T}_{x}c^{i}_{\text{core,cs,rp}}[\text{recog\_sentence}]$, $\xi \triangleq {}^{T}_{x}c^{i}_{\text{core,cs,da}}[\text{cmd}]$

3.8.17 Behaviours corresponding to the RAPP API functions

Below there are presented behaviours with the corresponding Rapp API functions:

- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,playAudio}}$ corresponds to the playAudio function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,textToSpeech}}$ corresponds to the textToSpeech function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,wordSpotting}}$ corresponds to the wordSpotting function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,captureAudio}}$ corresponds to the captureAudio function and to the captureAudio (with silence recognition) function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,voiceRecord}}$ corresponds to the voiceRecord function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveTo}}$ corresponds to the moveTo function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveVel}}$ corresponds to the moveVel function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,getRobotPosition}}$ corresponds to the getRobotPosition function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveStop}}$ corresponds to the moveStop function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveJoint}}$ corresponds to the moveJoint function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,takePredefinedPosture}}$ corresponds to the takePredefinedPosture function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,rest}}$ corresponds to the rest function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,moveAlongPath}}$ corresponds to the moveAlongPath function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,globalLocalization}}$ corresponds to the globalLocalization function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,lookAtPoint}}$ corresponds to the lookAtPoint function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,captureImage}}$ corresponds to the captureImage function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,setCameraParams}}$ corresponds to the setCameraParams function,
- Behaviour ${}^{c}_{+}\mathcal{B}_{\text{core,cs,getTransform}}$ corresponds to the getTransform function.

References

[1] C. Zieliński, T. Kornuta, and M. Boryń, "Specification of robotic systems on an example of visual servoing," in *10th International IFAC Symposium on Robot Control (SYROCO)*, vol. 10, 2012, pp. 45–50.

[2] T. Kornuta and C. Zieliński, "Robot control system design exemplified by multi-camera visual servoing," *Journal of Intelligent & Robotic Systems*, vol. 77, no. 3–4, pp. 499–524, 2015.

[3] C. Zieliński and T. Winiarski, "General specification of multi-robot control system structures," *Bulletin of the Polish Academy of Sciences – Technical Sciences*, vol. 58, no. 1, pp. 15–28, 2010.

[4] C. Zieliński, T. Kornuta, and T. Winiarski, "A systematic method of designing control systems for service and field robots," in *19-th IEEE International Conference on Methods and Models in Automation and Robotics, MMAR'2014*. IEEE, pp. 1–14.

[5] P. Trojanek, T. Kornuta, and C. Zieliński, "Design of asynchronously stimulated robot behaviours," in *Robot Motion and Control (RoMoCo), 9th Workshop on*, K. Kozłowski, Ed., 2013, pp. 129–134.

[6] C. Zieliński, W. Kasprzak, T. Kornuta, W. Szynkiewicz, P. Trojanek, M. Walęcki, T. Winiarski, and T. Zielińska, "Control and programming of a multi-robot-based reconfigurable fixture," *Industrial Robot: An International Journal*, vol. 40, no. 4, pp. 329–336, 2013.