Culturally Sensitive Social Robotics

for Africa

**D2.3 Visitor Behaviour Specification**

Due date: 3**1/12/2023**

Submission Date: **25/11/2023**

Start date of project: **01/07/2023**

Duration: **36 months**

Lead organisation for this deliverable: **Carnegie Mellon University Africa**

Responsible Person: **D. Vernon** Revision: **1.0**

|  |  |  |
| --- | --- | --- |
|  | **Project funded by the African Engineering and Technology Network (Afretec) Inclusive Digital Transformation Research Grant Programme** | |
|  | **Dissemination Level** | |
| **PU** | Public | **PU** |
| **PP** | Restricted to other programme participants (including Afretec Administration) |  |
| **RE** | Restricted to a group specified by the consortium (including Afretec Administration) |  |
| **CO** | Confidential, only for members of the consortium (including Afretec Administration) |  |

**Executive Summary**

Deliverable D2.3 is concerned with the identification of the visitor’s behaviour in the two use-case scenarios described in Deliverable D2.1: Use Case Scenario Definition, version 1. Sections 3 and 4 of D2.1 define the visitor actions and describe them in perceptual terms, from the perspective of the robot. The purpose of this deliverable is to abstract these robot perceptions of the visitor’s behaviour, and characterize them as perception primitives that can be used to determine the visitor’s behaviour in relation to the use-case scenario environment, e.g.,   
detectPerson(number\_detected, person\_descriptor[]).

Deliverable D2.3 provides the requirements for work package WP4 on robot sensing, complementing the detailed specification already provided in the work plan.

## **Contents**

[**1.** **The Basis for Behavior Definition** 4](#_Toc151624905)

[**2.** **Perception Primitive Descriptions** 9](#_Toc151624906)

[**Principal Contributors** 16](#_Toc151624907)

[**Document History** 17](#_Toc151624908)

# **1. The Basis for Visitor Behavior Definition**

Deliverable D2.1 Use Case Scenario Definition identifies the set of baseline robot perceptions that are invoked when engaging with a visitor. These perceptions represent the behaviour of the visitor, and the state of the robot’s environment, as perceived by the robot. They are summarized in Table 1 below.

Table 2 defines a set of abstract perception primitives, each of which encapsulates a subset of one or more of these baseline robot perceptions. Each perception primitive has a set of parameters that, suitably chosen, allows it to achieve one or more required baseline robot perceptions.

Table 3 maps the baseline robot perceptions to the corresponding abstract parameterized perception primitives.

Section 2 provides a more detailed explanation of each perception primitive. It specifies the primitive parameters, and it identifies the ROS nodes that call the primitive and the associated ROS topics that are used to read input for the primitive and write output from the primitive.

|  |
| --- |
| Detect mutual gaze  Face detection  Face localization  Person detection  Person distance estimation  Person localization  Sound detection  Speech event  Sound localization  Tablet PC event |

Table 1: The robot perceptions listed in Section 3 of Deliverable 2.1.

|  |
| --- |
| checkMutualGaze() |
| getBody(body\_x, y, z) |
| getEyeGaze(eye, x, y, z) |
| getEyes(eyeL\_x, y, z, eyeR\_x, y, z) |
| getFaces(<x, y, z>) |
|  |
| getHands(<x, y, z>) |
| getHead(head\_x, y, z) |
| getHeadGaze(<plane\_x, y, z>, x, y, z) |
| getHeadGaze(x, y, z) |
| getObjects(<x, y, z>) |
|  |
| getObjectTableDistance(object\_x, y, z, vertical\_distance) |
| getSoundDirection(threshold, azimuth, elevation) |
|  |
|  |
|  |
|  |
| identifyVoice(voice\_descriptor) |
| recognizeSpeech(text) |
| trackFace(seed\_x, y, z, time\_interval, projected\_x, y, z) |
| trackHand(seed\_x, y, z, time\_interval, projected\_x, y, z) |
|  |

Table 2: Perception primitives

|  |  |
| --- | --- |
| Active face tracking | trackFace(seed\_x, y, z, projected\_x, y, z) |
| Child body pose recognition | getBodyPose(<joint\_i>) |
| Compute child's eye gaze | getEyeGaze(eye, x, y, z) |
| Compute child's head gaze | getHeadGaze(x, y, z) |
| Detect mutual gaze | checkMutualGaze() |
| Detection of high amplitude sounds | getSoundDirection(threshold, azimuth, elevation) |
| Determine intersection of gaze and table | getHeadGaze(<plane\_x, y, z>, x, y, z) |
| Face detection | getFaces(<x, y, z>) |
| Face localization | getFaces(<x, y, z>) |
| Face recognition | identifyFace(x, y, z, face\_id) |
| Facial expression recognition | identifyFaceExpression(x, y, z, expression\_id) |
| Grip point localization | getGripLocation(object\_x, y, z, grip\_x, y, z) |
| Hand detection | getHands(<x, y, z>) |
| Hand localization | getHands(<x, y, z>) |
| Hand tracking | trackHand(seed\_x, y, z, projected\_x, y, z) |
| Hand-object (body) alignment detection | getArmAngle(left\_azimuth, elevation, right\_azimuth, elevation) |
| Hand-object (body) spatial relationship | getHands(<x, y, z>)  getBody(body\_x, y, z) |
| Hand-object (eye) occlusion detection | getHands(<x, y, z>)  getEyes(eyeL\_x, y, z, eyeR\_x, y, z) |
| Hand-object (head) spatial relationship | getHands(<x, y, z>)  getHead(head\_x, y, z) |
| Hand-object (picture) occlusion detection | getHands(<x, y, z>)  getObjects(<x, y, z>) |
| Hand-object (picture) spatial relationship | getHands(<x, y, z>)  getObjects(<x, y, z>) |
| Hand-object spatial relationship | getHands(<x, y, z>)  getObjects(<x, y, z>) |
| Head and hand tracking | trackHand(seed\_x, y, z, projected\_x, y, z)  trackFace(seed\_x, y, z, projected\_x, y, z) |
| Object (destination) detection | getObjects(<x, y, z>) |
| Object (destination) localization | getObjects(<x, y, z>) |
| Object (destination) recognition | identifyObject(x, y, z, object\_id) |
| Object (picture) detection | getObjects(<x, y, z>) |
| Object (picture) localization | getObjects(<x, y, z>) |
| Object (sand-tray) detection | getObjects(<x, y, z>) |
| Object (sand-tray) localization | getObjects(<x, y, z>) |
| Object tracking | trackObject(objectDescriptor, seed\_x, y, z, projected\_x, y, z) |
| Object-table spatial relationship | getObjectTableDistance(object\_x, y, z, vertical\_distance) |
| Search this area for a picture | getObjects(centre\_x, y, z, radius, <x, y, z>) |
| Sound localization in horizontal plane | getSoundDirection(threshold, azimuth, elevation) |
| Speech recognition | recognizeSpeech(text) |
| Trajectory classification | identifyTrajectory(<x, y, z, t>, trajectory\_descriptor) |
| Voice recognition | identifyVoice(voice\_descriptor) |

Table 3: Correspondence between robot perceptions and perception primitives.

# **2. Perception Primitive Descriptions**

In the following, the functionality of each perception primitive is defined. To make it easier to understand, parameters that pass values *to* the primitive are written in green whereas parameters that *return* values are written in red.

2.1 checkMutualGaze()

The checkMutualGaze() primitive returns a value of true or false depending on whether the child and the robot are looking at each other. Note, when detecting mutual gaze, the robot only has to determine whether or not the child looks at the robot's head, but not necessarily at the robot's eyes. The gaze has to be held for a minimum period, e.g. 3 seconds.

2.2 getArmAngle(left\_azimuth, left\_elevation, right\_azimuth, right\_elevation)

The getArmAngle() primitive returns the azimuth and elevation angles of the child’s upper left and right arms, defined with respect to the robot’s torso, i.e. in a child-centred frame of reference.

2.3 getBody(body\_x, body\_y, body\_z)

The getBody() primitive returns the location of the centre of the child’s body.

2.4 getBodyPose(<joint\_i>)

The getBodyPose() primitive returns a vector (denoted by < >) of joint angles describing the child’s current body pose. The joint angles correspond to a canonical representation of a body skeleton, yet to be defined.

2.5 getEyeGaze(eye, x, y, z)

The getEyeGaze() primitive returns the location in the world at which the child’s eye gaze is directed. The gaze direction is given by the line connecting the child’s eye and that location. Any point on that line could be the object of the child’s attention. The eye to be used is passed in the eye parameter (left or right).

2.6 getEyes(eyeL\_x, eyeL\_y, eyeL\_z, eyeR\_x, eyeR\_y, eyeR\_z)

The getEyes() primitive the locations of the child’s left and right eyes, specified in Cartesian coordinates in a world frame of reference.

2.7 getFaces(<x, y, z>)

The getFaces() primitive returns a vector (denoted by < >) of locations where faces are visible in the therapy environment (viewed from any of the therapy cameras). Each location is specified by a triple of Cartesian coordinates x, y, and z.

2.8 getGripLocation(object\_x, object\_y, object\_z, grip\_x, grip\_y, grip\_z)

The getGripLocation() primitive returns the location of a grip point of a play object in the therapy environment. The location of the object is passed to the primitive.

2.9 getHands(<x, y, z>)

The getHands() primitive returns a vector (denoted by < >) of locations where hands are visible in the therapy environment (viewed from any of the therapy cameras). Each location is specified by a triple of Cartesian coordinates x, y, and z.

2.10 getHead(head\_x, head\_y, head\_z)

The getHead() primitive returns the locations of the child’s head, specified in Cartesian coordinates in a world frame of reference.

2.11 getHeadGaze(x, y, z)

The getHeadGaze() primitive returns the location in the world at which the child’s head is directed. The head direction is given by the line connecting the mid-point between the child’s eyes and that location. Any point on that line could be the object of the child’s attention.

2.12 getHeadGaze(<plane\_x, plane\_y, plane\_z>, x, y, z)

This alternative version of the getHeadGaze() primitive returns the location on a flat surface (i.e. a plane defined by a vector of Cartesian coordinates ) at which the child’s head is directed. Typically, this plane corresponds to the table on which play objects are placed.

2.13 getObjects(<x, y, z>)

The getObjects() primitive returns a vector (denoted by < >) of locations where play objects are visible in the therapy environment (viewed from any of the therapy cameras). Each location is specified by a triple of Cartesian coordinates x, y, and z.

2.14 getObjects(centre\_x, centre\_y, centre\_z, radius, <x, y, z>)

This alternative form of getObjects() primitive returns a vector (denoted by < >) of locations where play objects are visible in the therapy environment (viewed from any of the therapy cameras). Each location is specified by a triple of Cartesian coordinates x, y, and z. In this case, the search for the objects is restricted to a circular region given by the centre coordinates and radius parameters.

2.15 getObjectTableDistance(object\_x, object\_y, object\_z, vertical\_distance)

The getObjectTableDistance() primitive returns the vertical distance of an object given by the Cartesian coordinates to the table on which play objects are places.

2.16 getSoundDirection(threshold, azimuth, elevation)

The getSoundDirection() primitive returns the horizontal (azimuth) and vertical (elevation) angles defining the direction to the loudest sound in the environment (e.g. when someone is speaking). The angles are defined with respect to the robot’s local frame of reference centred in its torso.

2.17 identifyFace(x, y, z, face\_id)

The identifyFace() primitive classifies the face at the specified location. The possible classes are determined during the set up phase. Typically, they will include the therapist and the children involved in the therapy sessions. The class identifier is returned in the face\_id parameter.

2.18 identifyFaceExpression(x, y, z, expression\_id)

The identifyFaceExpression() primitive classifies the expression of the face at the specified location. The possible classes are determined during the set up phase. Typically, they will be happy, sad, angry, or fearful (i.e. the four classes of emotion used in the interventions defined in Deliverable D1.1). The emotion identifier is returned in the expression\_id parameter.

2.19 identifyObject(x, y, z, object\_id)

The identifyObject() primitive classifies the object at the specified location. The possible classes are determined during the set up phase. Typically, they will correspond to a designated set of play objects, pictures to be placed on the table, or pictures on the sand-tray. The object identifier is returned in the object\_id parameter.

2.20 identifyTrajectory(<x, y, z, t>, trajectory\_descriptor)

The identifyTrajectory() primitive classifies the trajectory defined by a vector of 4-tuples, each 4-tuple defining a 3-D location and a time-stamp. The possible classes are determined during the set up phase. Typically, they will correspond to a designated set of hand gestures to be made by the child (e.g. a wave). The trajectory identifier is returned in the trajectory\_descriptor parameter.

2.21 identifyVoice(voice\_descriptor)

The identifyVoice() primitive classifies the voice that is currently speaking. The possible classes are determined during the set up phase. Typically, they will include the therapist and the children involved in the therapy sessions. The voice identifier is returned in the voice\_descriptor parameter.

2.22 recognizeSpeech(text)

The recognizeSpeech() primitive returns a textual representation of anything currently being spoken by the therapist or the child.

2.23 trackFace(seed\_x, seed\_y, seed\_z, time\_interval, projected\_x, projected\_y, projected\_z)

The trackFace() primitive tracks the face that is currently located at the position given by the Cartesian seed coordinates and returns the projected location in the next time interval. The duration of the time interval is specified in milliseconds. By reassigning the projected location to the seed and repeatedly invoking trackFace() the locations of the face can be tracked and recorded.

2.24 trackHand(seed\_x, seed\_y, seed\_z, time\_interval, projected\_x, projected\_y, projected\_z)

The trackHand() primitive tracks the hand that is currently located at the position given by the Cartesian seed coordinates and returns the projected location in the next time interval. The duration of the time interval is specified in milliseconds. By reassigning the projected location to the seed and repeatedly invoking trackHand() the locations of the hand can be tracked and recorded.

2.25 trackObject(objectDescriptor, seed\_x, seed\_y, seed\_z, time\_interval, projected\_x, projected\_y, projected\_z)

The trackObject() primitive tracks the object given by the object descriptor and currently located at the position given by the Cartesian seed coordinates and returns the projected location in the next time interval. The duration of the time interval is specified in milliseconds. By reassigning the projected location to the seed and repeatedly invoking trackObject() the locations of the object can be tracked and recorded.

# **Principal Contributors**

The main authors of this deliverable are as follows (in alphabetical order).

David Vernon, Carnegie Mellon University Africa.

# **Document History**

**Version 1.0**

First draft.  
 David Vernon.   
 24 November 2023.