CSSR for Africa

Culturally Sensitive Social Robotics for Africa

6.1 Use Case Implementation

Due date: **31/03/2025**
Submission Date: **18/04/2025**
Revision Date: **N/A**

Start date of project: **01/07/2023** Duration: **36 months**

Lead organisation for this deliverable: **Carnegie Mellon University Africa (for Wits)**

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 D6.1 documents the outcome of Task 6.1, i.e., the implementation of the two use cases defined in Work Package WP2, using the outcomes of WP1 - WP5, i.e., the cultural knowledge, the scenario specification, and the integrated robot's sensory and interaction capabilities. Specifically, the use cases are captured using the robot mission specification methodology documented in Deliverable D5.4.2 Robot Mission Language, i.e., using behavior trees [1, 2]. The resultant behavior tree provides the input to the `behaviorController` ROS node documented in Deliverable D5.4.3. Running the `cssr_system` ROS package against the behavior tree robot mission specification provides a demonstration of the complete working system for the corresponding use case.

In the CSSR4Africa work plan, this deliverable is assigned to the University of the Witswatersrand. However, the material in this report was developed and written by Carnegie Mellon University Africa. This was necessary because, due to extensive delays in the delivery of the Pepper robot to the University of the Witswatersrand, little or no progress had been made on three key inputs for Task 6.1 and Deliverable D6.1, viz. Deliverables D5.4.1 Cultural Knowledge Ontology & Culture Knowledge Base, D5.4.2 Robot Mission Language, and D5.4.3 Robot Mission Interpreter, all of which were assigned to the University of the Witswatersrand. Consequently, Carnegie Mellon University Africa took joint responsibility for these four deliverables (among others, specificially D5.5.2.1, D5.5.4, D6.2). Since this involved a significant amout of additional, unplanned effort, only one use case, the laboratory tour, has been implemented to date, leaving the second use case, the receptionist, to be implemented later.

Contents

1 Introduction

This deliverable provides a detailed walkthrough of the behaviour tree implementation the Lab Tour robot mission specification, as defined by the D2.1 Use Case Scenario. Section 2.1 introduces the generated XML output, which is the representation of robot mission specification behavior tree, by describing the organization of the XML file. Section 2.3 discusses the incorporation and structuring of subtrees, which modularize complex behaviors and enable reusability within the mission specification. Additionally, Section 2.4 explains mission nodes, which represent specific tasks or actions executed by the robot. Lastly, a visual representation is provided through the behavior tree diagram in Fig.1).

2 Behaviour Tree Implementation of Use Case Scenario 1: Laboratory Tour

This section details the implementation of a robot mission specification based on the operational guidelines provided in the "Lab Tour" scenario (see Deliverable D2.1 User Scenario Specification). The mission is structured as a behavior tree, which serves as the control architecture for coordinating the robot's actions. To design and visualize this behavior tree, we employ the Groot2 IDE, using its intuitive drag and drop mechanisms as outline above. The mission interpreter, implemented using the BehaviorTree.CPP library (described in Deliverable D5.4.3 Robot Mission Interpreter), executes the behavior tree within a ROS-based framework, forming an integral part of the CSSR4Africa software system. For an overview of the system architecture, please refer to D3.1 System Architecture.

2.1 XML Output

As explained above, the mission is designed using the Groot2 IDE. The Groot2 IDE allows for the design of behavior trees using a graphical interface. The mission specification represented as a behavior tree is then exported as an XML file, which is used by the mission interpreter to execute the mission. The XML representation of the behavior tree designed for the "Lab Tour" scenario is explained in the following sections. For the graphical representation of the behavior tree, refer to Figure 1.

The robot mission specification file is has three sections, the file root, the subtree specifications, and the mission nodes. These are described in the following sections.

2.2 File Root

The root element of the XML file is defined as:

```
<?xml version="1.0" encoding="UTF-8"?>
<root BTCPP_format="4" main_tree_to_execute="TourGuide">
<!-- .... -->
</root>
```

The attribute `BTCPP_format="4"` specifies that this behavior tree is built for version 4 of the BehaviorTree.CPP library. This is important because the robot mission interpreter must be implemented with the correct version, in this case, version 4, for the behavior tree specification to be executed correctly. It is worth noting that the Groot2 IDE supports both version 3 and version 4 of the library, configurable within the settings, so this attribute helps ensure compatibility between the designed behavior tree and the mission interpreter. The attribute `main_tree_to_execute="TourGuide"` designates `TourGuide` as the primary behavior tree to be executed, indicating which tree within the file serves as the entry point for the mission.

Fig.1 is illegible. I suggest you break it up into nine diagrams, the first with the root, sequence, and seven level-three nodes, the remaining six figures with the five subtrees for the five phases, and the subtree that contains the MaybeAnother-TimeSpeech node. It is not clear why this latter subtree isn't accorded a distinct phase and a shaded background. Neither is it clear why there is a sequence node outside the V EndTour subtree. It only has one exit arrow: how can it be a sequence? The remaining two figures should show the **_NavigateToLocation** and **_GoHome** subtrees.

2.3 Subtrees

To facilitate an efficient mission design process and enhance readability and clarity, the overall mission was structured into several distinct subtrees, each representing a logical segment of the overall task defined in the use case scenario. This enabled easier development and debugging and allows for future modifications. By organizing the behavior tree into modular, clearly defined subtrees, the complexity of mission specifications is significantly reduced. Then, by connecting these modular subtrees in a well-defined logical sequence, the final behavior tree is constructed, which serves as the complete mission specification for the use case scenario as defined in Deliverable D2.1 Use Case Scenario.

2.3.1 TourGuide Subtree

The TourGuide subtree is the main behavior tree that orchestrates the robot's actions during the tour. It is divided into five segments, each representing a distinct phase of the tour experience. The segments are executed sequentially, with the robot transitioning from one segment to the next based on the outcome of the previous segment. The segments are defined as follows.

```
<BehaviorTree ID="TourGuide">
  <Sequence>
    <SubTree ID="I. DetectVisitor"/>
    <SubTree ID="II. EngageVisitor"/>
    <SubTree ID="III. QueryVisitorResponse"/>
    <Fallback>
      <Sequence>
        <Inverter>
          <IsVisitorResponseYes/>
        </Inverter>
        <MaybeAnotherTimeSpeech/>
      </Sequence>
      <Sequence>
        <SubTree ID="IV. VisitLandmark"/>
        <SubTree ID="V. EndTour"/>
      </Sequence>
    </Fallback>
  </Sequence>
</BehaviorTree>
```

2.3.2 I. DetectVisitor Subtree

The DetectVisitor subtree is dedicated to identifying when a visitor is present. It utilizes various sensors available to the robot to continuously monitor its surroundings. The robot is animated to appear lively, actively scanning its environment to reliably localize and track a visitor before initiating an interaction.

I do not like the use of Roman ordinal numbers in the behaviour tree identifiers, e.g., `I. DetectVisitor`. I suggest we remove them. I realize they are intended to indicate the phase of the tour but it seems to me they are unnecessary.

```
<BehaviorTree ID="I. DetectVisitor">
<Sequence>
  <Parallel failure_count="2" success_count="2">
    <Fallback>
      <EnableAnimateBehavior />
      <HandleFallBack />
    </Fallback>
    <Fallback>
```

```
          <ScanningOvertAttentionMode/>
          <HandleFallBack />
        </Fallback>
      </Parallel>
      <IsVisitorDiscovered />
    </Sequence>
  </BehaviorTree>
```

2.3.3 II. EngageVisitor Subtree

Ibid.

Once a visitor is detected, the robot transitions to actively engaging them. In this phase, the robot makes a welcoming gesture, greets the visitor verbally, and introduces itself as the tour guide. The engagement involves adjusting its body language to establish a friendly and approachable interaction.

```
<BehaviorTree ID="II. EngageVisitor">
<Sequence>
  <Fallback>
    <Sequence>
      <DisableAnimateBehavior />
      <WelcomeGesture />
      <EnableAnimateBehavior />
    </Sequence>
    <HandleFallBack />
  </Fallback>
  <Fallback>
    <WelcomeSpeech />
    <HandleFallBack />
  </Fallback>
  <Fallback>
    <SeekOvertAttentionMode />
    <HandleFallBack />
  </Fallback>
  <IsMutualGazeDiscovered />
  <Fallback>
    <SocialOvertAttentionMode />
    <HandleFallBack />
  </Fallback>
  <Fallback>
    <QueryTourSpeech />
    <HandleFallBack />
  </Fallback>
</Sequence>
</BehaviorTree>
```

2.3.4 III. QueryVisitorResponse Subtree

Ibid.

After the initial greeting, the robot seeks confirmation from the visitor about whether they would like to take the tour. This segment handles both speech-based and tablet-based responses. The robot asks a clear question, listens for a “Yes” or “No” response (using either automatic speech recognition or a visual touch interface), and prompts repeatedly if the response is unclear. If the response is positive, the robot proceeds to the next segment. Otherwise, it provides a polite response and ends the interaction.

```
<BehaviorTree ID="III. QueryVisitorResponse">
<Fallback>
  <Sequence>
    <IsASREnabled />
    <RetryUntilSuccessful num_attempts="3">
      <Sequence>
        <Fallback>
          <SayYesNoSpeech />
          <HandleFallBack />
        </Fallback>
        <IsYesNoUttered/>
      </Sequence>
    </RetryUntilSuccessful>
  </Sequence>
  <Fallback>
    <IsASREnabled />
    <RetryUntilSuccessful num_attempts="3">
      <Sequence>
        <Fallback>
          <PressYesNoSpeech />
          <HandleFallBack />
        </Fallback>
        <Fallback>
          <PressYesNoDialogue />
          <HandleFallBack />
        </Fallback>
      </Sequence>
    </RetryUntilSuccessful>
  </Fallback>
</Fallback>
</BehaviorTree>
```

2.3.5 IV. VisitExhibit Subtree

Ibid.

With a positive response, the tour moves into the exhibit visit segment. Here, the robot guides the visitor from one exhibit to another. For each exhibit, the robot retrieves and announces information about the exhibit from a knowledge base, navigates to the location, checks for visual contact to verify continuation, uses gestures such as pointing to highlight key aspects of the exhibit, and provides descriptive commentary about what is being shown. This segment is designed to be repeatable for each exhibit along the tour route.

```
<BehaviorTree ID="IV. VisitExhibit">
<Sequence>
  <Fallback>
    <RetrieveListOfExhibits />
    <HandleFallBack />
  </Fallback>
  <Inverter>
    <KeepRunningUntilFailure>
      <Sequence>
        <IsListWithExhibit />
        <Sequence>
          <Fallback>
            <SelectExhibit />
            <HandleFallBack />
          </Fallback>
          <Fallback>
            <FollowMeSpeech />
            <HandleFallBack />
          </Fallback>
          <SubTree ID="_NavigateToLocation" />
          <Fallback>
            <DescribeExhibitSpeech_1 />
            <HandleFallBack />
          </Fallback>
          <Fallback>
            <PerformDeicticGesture name="Point to Exhibit" />
            <HandleFallBack />
          </Fallback>
          <Fallback>
            <DescribeExhibitSpeech_2 />
            <HandleFallBack />
          </Fallback>
        </Sequence>
      </Sequence>
    </KeepRunningUntilFailure>
  </Inverter>
</Sequence>
</BehaviorTree>
```

2.3.6 V. EndTour Subtree

Ibid.

The final segment concludes the tour experience. Once all exhibits have been visited, the robot escorts the visitor back to the entrance. It communicates that the tour has ended, expresses gratitude and hope

that the visitor enjoyed the tour, and finally says goodbye while performing a farewell gesture. This segment ensures a polite and complete wrap-up of the interaction.

```
    <BehaviorTree ID="V. EndTour">
    <Sequence>
      <Fallback>
        <EndTourSpeech />
        <HandleFallBack />
      </Fallback>
      <Fallback>
        <LookUpEntrance />
        <HandleFallBack />
      </Fallback>
      <SubTree ID="_NavigateToLocation" />
      <Fallback>
        <HereIsTheDoorSpeech />
        <HandleFallBack />
      </Fallback>
      <Fallback>
        <PerformDeicticGesture name="Point to Exit" />
        <HandleFallBack />
      </Fallback>
      <Parallel failure_count="1" success_count="2">
        <Fallback>
          <GoodbyeGesture />
          <HandleFallBack />
        </Fallback>
        <Fallback>
          <SayGoodBye />
          <HandleFallBack />
        </Fallback>
      </Parallel>
      <SubTree ID="_GoHome" />
    </Sequence>
  </BehaviorTree>
```

Two additional subtrees, **_NavigateToLocation** and **_GoHome**, are defined to leverage modularity and reusability, fundamental benefits inherent in the behavior tree approach. Unlike the previously discussed subtrees, which primarily served to segment the mission into clear, readable chunks, these subtrees encapsulate behaviors that occur repeatedly throughout the mission. Abstracting these common behaviors into separate subtrees significantly reduced redundancy, enhances maintainability, and ensures consistency across the overall behavior tree structure. As a direct result, the mission specification becomes easier to design and maintain. This modular design is a case in point of one of the main strengths of behavior trees, which is enabling efficient reuse of behavior definitions within complex robotic missions.

Repeatedly? **_NavigateToLocation** appears twice but **_GoHome** only appears once in the overall behavior tree. I guess it could appear more often in the case of failure handling.

2.3.7 _NavigateToLocation Subtree

The main mission node in this subtree is the **Navigate** action node. This node is the one that's directly responsible for navigating the robot to a specified location. The robot uses its localization and mapping capabilities to plan a path to the target location and execute the navigation. But, before the robot navigates to a location, it must first disable **overt attention mode** and the **animate behavior subsystem** (if it has been enabled). This is necessary since the robot doesn't need to set its gaze anywhere but right in front of it or perform any gestures while navigating. The robot must focus on the navigation task and ensure it reaches the target location successfully. Once it has reached its destination, the robot must re-enable the overt attention mode to resume its interaction with the visitor. This set of behaviors that accompany the navigation task in almost every instance, have been encapsulated in the `_NavigateToLocation` subtree.

Neither do I like the use of the leading underscores in the behaviour tree identifiers. What do they signify? Unless there is a good reason for using them, I suggest we remove them.

```
<BehaviorTree ID="_NavigateToLocation">
  <Sequence>
    <Fallback>
      <DisabledOvertAttentionMode />
      <HandleFallBack />
    </Fallback>
    <Fallback>
      <DisableAnimateBehavior />
      <HandleFallBack />
    </Fallback>
    <Fallback>
      <Navigate />
      <HandleFallBack />
    </Fallback>
    <Fallback>
      <SeekOvertAttentionMode />
      <HandleFallBack />
    </Fallback>
    <IsMutualGazeDiscovered />
  </Sequence>
</BehaviorTree>
```

2.3.8 _GoHome Subtree

Ibid.

Similar to _NavigateToLocation, the _GoHome subtree encapsulates behaviors that are common everytime the robot needs to navigate to the Home location. The coordinates of that location are predefined and stored in the Environment Knowledge Base. The robot must first look up the home location, disable overt attention mode, and the animate behavior subsystem. Only then does it navigate to the home location. Once it has reached the home location, unlike the _NavigateToLocation subtree, the robot doesn't need to re-enable the overt attention mode.

```
<BehaviorTree ID="_GoHome">
<Sequence>
  <Fallback>
    <LookUpHome />
    <HandleFallBack />
  </Fallback>
  <Fallback>
    <DisabledOvertAttentionMode />
    <HandleFallBack />
  </Fallback>
  <Fallback>
    <DisableAnimateBehavior />
    <HandleFallBack />
  </Fallback>
  <Fallback>
    <Navigate />
    <HandleFallBack />
  </Fallback>
</Sequence>
</BehaviorTree>
```

2.4 Mission Nodes

The leaf nodes, which include both action and condition nodes, are where the custom functionality is implemented. Combined with control flow nodes, these building blocks enable the definition of the desired behaviors. A total of 33 action and condition nodes were defined for the "Lab Tour" scenario, with many of these nodes reused multiple times throughout the behavior tree. These custom nodes are comprehensively listed in Table 1, which provides the name, type, and description of each node. Note that the actual logic and implementation of these nodes are not detailed here; they are encapsulated within the robot mission interpreter, which executes the behavior tree, as described in Deliverable D5.4.3 Robot Mission Interpreter.

Table 1: High-Level Mission Node Descriptions

Node	Type	Description
DescribeExhibitSpeech_1	Action	Delivers the first part of an auditory description of the current exhibit to inform visitors about its details.
DescribeExhibitSpeech_2	Action	Delivers the second part of an auditory description of the current exhibit to inform visitors about its details.
DisableAnimateBehavior	Action	Disables the robot's animation behaviors
DisabledOvertAttentionMode	Action	Deactivates overt attention behaviors

Node	Type	Description
EnableAnimateBehavior	Action	Activates the robot capability to have the appearance of an animate agent
EndTourSpeech	Action	Delivers the concluding remarks of the tour, signaling the end of the visit.
FollowMeSpeech	Action	Instructs visitors to follow the robot, guiding them to the next segment of the tour.
GoodbyeGesture	Action	Executes a farewell gesture, visually marking the end of the interaction.
HandleFallBack	Action	Acts as a contingency mechanism to manage unexpected failures during mission execution
HereIsTheDoorSpeech	Action	Informs visitors about the location of an exit or transition point within the environment.
IsASREnabled	Condition	Evaluates whether the system's speech recognition feature is active, influencing subsequent interactive behaviors.
IsListWithExhibit	Condition	Determines if there are remaining exhibits to visit
IsMutualGazeDiscovered	Condition	Assesses whether mutual gaze with a visitor is established, a key element for engaging interactions.
IsVisitorDiscovered	Condition	Detects the presence of a visitor
IsVisitorResponseYes	Condition	Checks for an affirmative response from the visitor
IsYesNoUttered	Condition	Verifies whether a clear yes/no response has been provided
LookUpEntrance	Action	Accesses the location data for the entrance
LookUpHome	Action	Fetches the coordinates of the "Home" location
MaybeAnotherTimeSpeech	Action	Communicates a polite postponement of the tour
Navigate	Action	Initiates navigation by directing the robot toward a specified location within the environment.
PerformDeicticGesture	Action	Executes a pointing gesture to direct the visitor's attention to a specific exhibit or location.
PressYesNoDialogue	Action	Starts a binary (Yes/No) dialogue with the visitor to capture their input
PressYesNoSpeech	Action	Prompts the visitor verbally for a yes/no response
QueryTourSpeech	Action	Invites the visitor to participate in the tour, thereby initiating the interactive experience.
RetrieveListOfExhibits	Action	Gathers a list of exhibits to be visited, forming the basis for sequencing of the tour.
SayGoodByeSpeech	Action	Delivers a farewell message to mark the end of the tour
SayYesNoSpeech	Action	Clearly instructs the visitor to provide a yes/no response, reinforcing the expected interaction format.
ScanningOvertAttentionMode	Action	Switches the robot's focus to a scanning mode, enabling it to search for and assess visitor presence.
SelectExhibit	Action	Chooses the next exhibit for the tour
SocialOvertAttentionMode	Action	Engages a social attention mode that enhances the robot's interactive presence

Node	Type	Description
START_OF_TREE	Action	Marks the beginning of the mission, serving as a logical anchor for debugging and tracking mission progression for the robot mission interpreter.
WelcomeGesture	Action	Executes a welcoming gesture
WelcomeSpeech	Action	Delivers an initial welcome message to engage the visitor at the start of the tour.

2.5 Behavior Tree Diagram

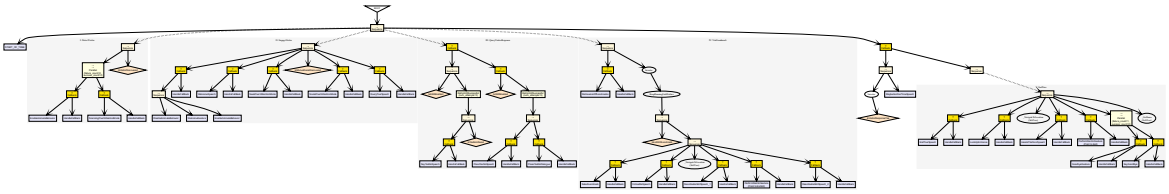

Figure 1: Behavior Tree Diagram of the Robot Mission Specification for the “Lab Tour” Scenario

References

[1] R. Ghzouli, T. Berger, E. B. Johnsen, A. Wasowski, and S. Dragul. Behavior trees and state machines in robotics applications. *IEEE Transactions on Software Engineering*, 49(9):4243 – 4267, 2023.

[2] E. Dortmans and T. Punter. Behavior trees for smart robots practical guidelines for robot software development. *Journal of Robotics*, 2022.

Principal Contributors

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

Tsegezeab Tefferi, Carnegie Mellon University Africa.
David Vernon, Carnegie Mellon University Africa.

Document History

Version 1.0

First version created by moving and reorganizing Section 3 from Deliverable D5.4.2.
David Vernon.
18 April 2025.