



# FROG

FUN ROBOTIC  
OUTDOOR GUIDE

## **Deliverable: D2.4**

### **AR Robot Application component**

#### **Consortium**

UNIVERSITEIT VAN AMSTERDAM (UVA)  
YDREAMS - INFORMATICA S.A. (YD)  
IDMIND - ENGENHARIA DE SISTEMAS LDA (IDM)  
UNIVERSIDAD PABLO DE OLAVIDE (UPO)  
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE (ICL)  
UNIVERSITY OF TWENTE (UT)

Grant agreement no. **288235**

Funding scheme **STREP**



Imperial College  
London

UNIVERSITEIT TWENTE.

## DOCUMENT INFORMATION

<b>Project</b>	
Project acronym:	FROG
Project full title:	Fun Robotic Outdoor Guide
Grant agreement no.:	288235
Funding scheme:	STREP
Project start date:	1 October 2011
Project duration:	36 Months
Call topic:	ICT-2011.2.1 Cognitive Systems and Robotics (a), (d)
Project web-site:	www.frogrobot.eu

<b>Document</b>	
Deliverable number:	D2.4
Deliverable title:	AR Robot Application component
Due date of deliverable:	M23 – 31 August 2013
Actual submission date:	M23 – 29 August 2013
Editors:	YD, UT
Authors:	YD, UT
Reviewers:	UT
Participating beneficiaries:	2,3,4,5,6
Work Package no.:	2
Work Package title:	Robot Control, Navigation and Location Based Content
Work Package leader:	UPO
Work Package participants:	2,3,4,5,6
Estimated person-months for deliverable:	15
Dissemination level:	Public
Nature:	Other
Version:	Final -revision
Draft/Final:	Draft
No of pages (including cover):	39
Keywords:	Augmented Reality, robot software frameworks

## D2.4 – Revised - AR Robot Application component

### Addendum – Revision - M36.

This deliverable describes the implementation details of the Augmented Reality module of the FROG robot and includes some evaluation results of the performance in July 2014.

YDreams wrote the original version of the deliverable in August 2013. At the end of December 2013 (M27), YDreams left the project and in the second half of April 2014 (M31) Randy Klaassen and Jan Kolkmeier (both from UT) took on the work of implementing an Integration Framework and AR Content for the FROG project.

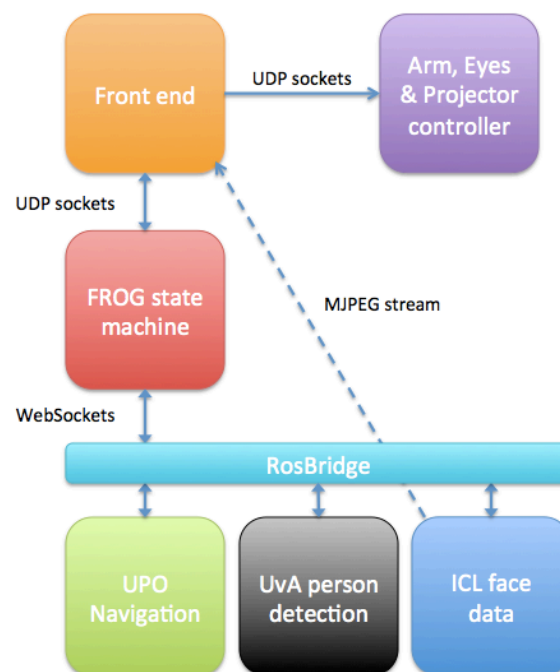
This revision will describe the final implementations and new solutions that have been adopted in bringing “interaction” and “fun” to the FROG.

### Summary

Strictly speaking, the FROG AR module includes a lot of multimedia content. Real Augmented Reality (a view of the real world supplemented by computer-generated sensory input such as sound, video or graphics) has, however, been implemented in several instances in the Royal Alcázar tour. *Please see chapter 2 of this addendum.*

The behaviour and decisions of the FROG robot have now been implemented as a State Machine running on the FROG’s Interaction Computer. The *Front end* is now responsible for playing content on the FROG robot and has been developed using the Unity platform - a cross-platform game creation system. The arm (antenna), eyes and projector of the FROG are driven by their own controllers and are controlled by the *Front end*.

This revised document is mostly about the *Front end*.



### 1. Introduction

Presentation of FROG’s content is mostly location related because the FROG’s mission is to navigate a tour route while giving information. Part of the content is triggered or adapted according to incoming

data about the users, their presence or absence, their estimated engagement level or, sometimes, their direct input via the touchscreen.

The FROG still has two image output devices, a video projector and a touchscreen.

The FROG's antenna is now certainly being used to point to artefacts and facilities (for example, indicating architectural features while describing the surroundings, and pointing out where the toilets are on passing). YDreams mentioned this but did not intend to implement it. A nice example was in September 2014 when the robot announced, relevantly: "These toilets are being refurbished" in passing. The antenna no longer houses a laser pointer but now has a webcam built in that plays an important role in the implementation of AR overlays, giving real-time feedback to users and as an integral part in various interactions with the user. The antenna is sometimes referred to as the snorkel.

The FROG can play sounds and videos. The touchscreen is used as sparingly as possible for user interaction relating to the tour itself – in fact, only at the beginning of the tour where the visitors are asked whether they are possibly interested in taking a tour and, after a short explanation of what they can expect, whether indeed they want to join a tour.

However, the specific advantages of a touchscreen as input device are seized upon for other user interaction – a quiz in the Vault room and some small ad hoc interactive games intended for amusing children of all ages (Figure 1).



**Figure 1** Children having fun with FROG at one of the Points of Interest during a tour

FROG now gives information about the tour process itself and about its assessment of the visitors. A timeline of the tour is shown as a simple straight route with relative distances between Points of Interest (POIs). This timeline changes colour progressively along the route as the tour progresses. An estimation of the user's engagement level is shown on a dial coloured from cool to warm (very interested to not really interested). The information for this output comes from the ICL human affective signals analyser. Users are invited to come closer to try out this FROG ability: feedback is given in the form of the engagement level dial, the camera view of the ICL camera and verbal (while asking the user to take a guided tour). After 30 seconds the final engagement level will be used to invite the user to take a guided tour. If the engagement level of the user seems to be high, FROG will invite the user for a guided tour by saying *"It seems you are very interested in FROG! Do you want to join us on a guided tour?"* If the engagement level is low, FROG will invite the user for a guided tour

by saying “It seems you are not really interested in FROG. But maybe we measured wrong. Do you want to join us on a guided tour?”

At Points of Interest 1, 2 and 3 data from the ICL human affective signals analyser (see D3.2) is used to decide how much detail will be presented about the POI during the tour. If the user(s) seems to be engaged FROG will present more details about the POI. If the user(s) seem not to be engaged FROG will skip these details and continue with the rest of the tour. Figure 2 presents an example of the use of the ICL human affective signals analyser at POI2. The users were engaged and FROG presented details about the history of the theatre at the Alcázar. The main window shows the camera view of the antenna, including the AR overlay. The bottom right window shows the camera view of the ICL human affective signals analyser, including the engagement level dial. The bottom left window presents the tour guide.



Figure 2 An example of the use of the ICL human affective signals analyser. In the bottom right corner the camera view and engagement level dial during one of the Points of Interest (engagement level is high). Face tracking and augmented reality is used to place overlays of masks over the faces of the persons in front of the robot

## 2. AR and localized content

The FROG State Machine now keeps track of the current location of the FROG robot, which makes it possible to start content and change the behaviour of FROG while navigating in certain areas or interesting points between two Points of Interest (POI). Three types of location-based adaptation of the behaviour of FROG or content have been implemented.

The behaviour of FROG will be adapted when the robot is in range of a Point of Interest to inform the users that it will soon finish navigating. FROG stops saying “navigating to next location, please follow me”. Users will know that a new point of interest has almost been reached so they can be prepared for the start of the content when the POI is reached.

Another feature that can be adapted based on the location is the volume of FROG's audio. When FROG is indoors (near the docking station in the shop (volume is set to 50%) and in the Tapestry Room and Vault Room (volume is set to 80%)) it lowers its volume level so as not to disturb or bother other visitors, clients and (human) tour guides.

A last feature that has been implemented based on the location of the FROG is the start of special content around places that can be useful for the visitors to know. FROG will tell the visitors where they can find the toilets, shop and bar while driving near these places.

### **3. Antenna**

By February 2014 the FROG antenna had evolved into a 3DOF arm with a green laser pointer surrounded by green LEDs mirroring those around the eyes.



*Antenna with laser pointer and LEDs*

During the integration meeting in June 2014 (M33) a simple webcam was mounted on the antenna with duct tape to test an idea. In the middle of August 2014 the UT Campus robot arrived in Twente with a Microsoft LifeCam Studio webcam mounted inside the antenna. This could be used to test application of the antenna webcam in simulation as the same model was also being built into the FROG's antenna.

The antenna is now used in the Seville mission:

- as a pointing or gesturing device
  - to point at something specific: e.g. pointing up at ceilings and showing overlays of architectural features on what the antenna camera relays via the touchscreen
  - to gesture in a general direction: e.g. a sweeping gesture to indicate the triple-arched wall in the Lion's Courtyard
- to show as-well-as tell its intentions: e.g. FROG says "looking for interested visitors" it hovers on individuals close by (based on person detection), showing what the antenna camera is picking up via the touchscreen
- for overlays in user interaction
  - dressing visitors in period costumes
  - placing theatrical masks on visitors' faces
  - virtual graffiti
  - ...
- for affective expressive content





*Testing the webcam temporarily mounted on the antenna*

#### **4. Front end Modalities & Controls used during tour.**

##### Screen/GUI:

- Main: Several different screens overlaying each other, sometimes “mixed”, from top to bottom (see “Media” for individual details):
  - [Interface layer (green frame around everything )
  - Image Layer
  - Video Plane
  - FaceOverlay Plane
  - Arm Webcam Stream
    - The live-stream of the Arm webcam
  - Black background.
- Timeline:
  - A timeline with the different epochs of the Alcazar (from ca. 600 to 2000 AD). Epochs use are colored by a consistent color code as used throughout the entire tour content.
  - Can be hidden, shown, and the position of the “needle” to mark a time can be changed.
- Tour Progress:
  - A progress bar with marks for start/introduction, POI’s and End of tour. The bar fills up during the tour (discrete time steps!).
- Status Left:
  - Tour guide Avatar
  - Status Icon (status of current robot state (navigating, looking for visitors, approaching, loading data, ...)
  - “Social QR 1”: FROG Twitter URL encoded in QR.
  - None/Idle
- Status Right:
  - MiniMap of Alcázar (with marker of the robot position in the middle)
  - Mini stream of ICL Camera
    - Including the “gauge” to show the engagement
  - Mini stream of Arm Webcam
  - Score count (used for Quiz at POI 5)

- “Social QR 2”: FROG Facebook URL encoded in QR.
- None/Idle

#### Media:

- Arm Webcam Stream:
  - Hide/show: ...
- AR Content:
  - A single/group of elements placed in the World COS can be enabled or disabled.
  - This is done in parallel with the respective “Look At” targets for the ARM
    - I.e.: looking at the “tapestry” target will also enable all AR-content that is connected to that target.
    - (see Hardware/Arm)
- Face/Body Overlay:
  - Set: change face overlay
  - Hide/show disables/enables rendering
  - Start/stop face tracking: starts/stops face tracking on image stream of the Arm webcam.
  - Show/hide body overlay: to show/hide the “traditional attire” (used in ad-hoc POI)
- Video:
  - Set video: ...
  - Play/Stop video (no pause!): ...
  - Hide/show video: ...
- Image:
  - Show an image: May contain transparency to overlay over other Media, such as webcam live stream or video
  - Hide image: ...
- Audio
  - Play music clips (as part of the tour)
  - See Avatar and Guide
  - Volume of audio is automatically controlled using marked areas on the map.

#### “Avatar” (frog representation on screen):

- Talk: says something with TTS (pre-generated, not on-line!), including “subtitles”
- Beep: non-lexical (is that the right word?) utterance to express robot activity (can be “positive” or “negative”)
- Mouth (show/hide): Mouth is usually rendered on the screen and with a simple (amplitude based) lip-sync. It is not rendered while talking when it would occlude important content on the screen. We have a happy/sad mouth, and some additional “funny faces” for the photo-pose-mode (end of tour).
- Eyes & Snorkel: See under Hardware.

#### Guide (the tour guide):

- Talk: Say a pre-recorded text fragment (usually part of the tour)
- Switch language: ...
- Show/hide: Also has an Avatar which we can show/hide in the smaller status areas
  - The Avatar has a blinking behaviour & lip-sync (amplitude based)

#### Hardware

- Control Eyes
  - Animation: Happy, Sad, Selfie, Band, ... (including speed of animation)
  - Brightness: Dim, Bright
- Control Arm (“Snorkel”)
  - Point directions in Robot Coordinate System: North, East, NE, NNE, ...
  - Point to fixed objects in World Coordinate System: Toilets, Lions Courtyard, “Gothic Cross Arcs”
  - Point to a “dynamic” object in Robot Coordinate System: Used to point Arm at a person. (
    - > request randy for position of person -> move dynamic target -> look at target)
  - Enable/disable Arm LED: whenever we point somewhere, we enable the Arm LED. When



Snorkle goes to “REST” state (for driving), we disable Arm LED.

- Control Projector:
    - Power off projector: ...
    - Power on projector: Typically used in combination with “blank screen”, to warm up projector before “Switch to Projector”.
    - Blank screen on/off: make the projector image (not) black while it is turned on.
    - Switch to Projector: Disable front screen and render image on projector instead.
- Switch to Screen: Disable projector and render image on screen instead.

## ***5. Evaluation of user experience of interaction strategies***

Earlier in the project an evaluation of the AR application performance and the effectiveness of the interaction strategies was performed. However due to failure of some devices the evaluation was insufficient. At this moment in the project (M36) we have done several studies on visitor experience of following the FROG tour, with invited visitor, naïve visitors and tour guides. We briefly present the results of the performance and effectiveness of the interaction strategies of these studies. This evaluation is based on the results of the studies performed in May 2013, February 2014, May 2014, June 2014 and September 2014 in the Royal Alcázar. We have presented more elaborate results of these studies in D4.1c and D4.1d.

In general the reactions to a robot as a guide are mixed, some visitors prefer a real guide, while others like the robot, because it is an easy way to obtain information about the site and does not take as long as the tours of the tour guides. Some visitors mentioned that they were unsure about following the robot as it looked as the appearance was designed for children. Children liked the robot a lot, however some were scared for a while, and some parents told that the robot was nice to follow for kids, because it was interactive and fun. The visitors that were interviewed during (one of) these studies (in total 5 semi-structured interviews with invited participants and in total 52 short semi-structured interviews) followed the whole or a part of the tour, so most of them were enthusiastic about the robot tour guide.

### ***Orientation of the robot***

In the study in May 2013 we evaluated the how the visitor engagement was influenced by the robot orientation. Visitors liked best when the robot was oriented with its front towards them and with its back towards the exhibit, because this gave a feeling of being seen by the robot. However, people stood closer to the robot and knew better where to look when the robot was oriented towards the exhibit. In follow-up studies we solved this issue by making use of the pointer. When the robot reached its stop, before it should start the information, the robot should “look” for the visitors, otherwise it feels as if the robot is ignoring them.

### ***Use of voice to offer information***

During the studies mentioned, several different voices were used for the robot. In the end a more mechanical voice was chosen for the robot and a voice-over to offer the information about the site. Visitors did not like a too mechanical voice to hear the information, as it was difficult to understand. The use of two voices was a solution that the visitors understood well. It made clear that the robot could not answer questions, as the voice-over was clearly a pre-recorded script. In the evaluation we found that visitors did not talk to the voice-over, but they did talk to the robot voice. From the guides we learned that the voice of the robot and the voice-over were too loud, especially in the indoor spots, which made it difficult for them to reach their own group.

### ***Use of the screen***

Lots of the information offered by FROG was presented on the screen. Visitors liked to get a better view of how things were ages ago. However, some of the visitors indicated that they would have liked to look more at the environment, as that it is the place it is all about.

### ***Use of projection***

Visitors liked the projection, especially when more than a few visitors stood around the robot, which gave everyone a chance to see the presentation. However, the place for projection needs to

be carefully chosen. When some obstacles are at the place of the projection (such as a bench) it is difficult for visitors to figure out the robot was projecting something.

#### ***Use of pointer***

The pointer was used for directing the gaze of the visitors, and most visitors understood that function early in the tour. Actually, when one person saw the pointer pointing in a certain direction, and that person started looking there, the others would also see the cue and started looking towards the indicated Point of Interest. Some of the participants in the study of June 2014 also mentioned that they liked the pointer, as it was used for making contact between the robot and the visitors.

#### ***Use of augmented reality***

Augmented reality was, for example, used in the Hall of Tapestries and placed flags on a map on the screen that was as displayed an image. By adding the flags, FROG explained which countries were visible on the map and it was easier for the visitors to see that the map was (according to our conventions) up-side-down. People liked to see the trick of the robot. Other moments the augmented reality was used, were to show what was behind the door of the Hall of Justice, to show the differences between the baroque and gothic styles in the Gothic Palace, to apply masks on peoples' faces when the robot told about the theater that had been in the Lion's Courtyard. The augmented reality was also used for fun moments, such as dressing up the visitors with clothes the Kings of Spain wore in ancient times and placing a graffiti tag on one of the walls of the Royal Alcázar. However, the guides let us know that they could imagine that the robot could show how a room was decorated, because now the rooms are all empty.

#### ***Use of different languages***

For the tours in September 2014, visitors were able to choose from three languages; English, Spanish and Dutch. We observed several times that visitors changed the language. One drawback was found when visitors left the robot as it performed the tour in Spanish when no more Spanish speaking visitors were around. Then visitors did not show so much interest in the robot because they did not understand it. Also, visitors tried to change the language during the navigation tasks, which influenced the performance of the robot.

#### ***Use of a quiz***

During the studies performed in June 2014, most visitors stated they liked the quiz best, because it made the tour interactive.

#### ***Use of face/emotion recognition***

It was difficult to understand for visitors why exactly the robot tracked them. They would understand better and behave/orientate in a useful manner for the face/emotion tracking when the robot would be more explicit about the use of the face/emotion recognition. Another problem of the face recognition was that it is fixed, but as visitors wanted to give all visitors around the robot some space to watch the robot, sometimes none of the faces of the visitors were tracked, resulting in the robot shortening the story while the visitors were interested.

To conclude we learned that the interviewed visitors mostly liked the robot, because it was easy and nice way to get information about the site and FROG was fun to follow. From observation we learned that visitors really like to have a picture of the robot or a picture of one of their friends posing with the robot. Not all visitors liked the robot; sometimes because it did not speak their language or because they had limited time to visit the Royal Alcázar.

## Table of Contents

<b>SUMMARY .....</b>	<b>5</b>
<b>1. INTRODUCTION .....</b>	<b>6</b>
<b>2. AR AND LOCALIZED CONTENT .....</b>	<b>7</b>
<b>3. ANTENNA .....</b>	<b>8</b>
3.1 As a pointing device .....	8
3.2 As an expression component .....	8
<b>4. INTEGRATION IN THE ROBOT FRAMEWORK .....</b>	<b>9</b>
4.1 The Content Layer .....	9
4.2 Smart Objects .....	9
4.3 Behaviors and Routines .....	10
4.4 Relationships between types .....	11
<b>5. MANAGING CONTENT.....</b>	<b>12</b>
5.1 Content Files .....	12
5.2 Locations .....	12
5.3 Content Manager .....	13
<b>6. PARTICULAR COORDINATE SPACE .....</b>	<b>14</b>
<b>7. EVALUATION OF THE AR APPLICATION PERFORMANCE AND THE EFFECTIVENESS OF THE INTERACTION STRATEGIES .....</b>	<b>18</b>
<b>ANNEX 1 - TYPES IN AR AND IN AR-BEHAVIOR RELATIONSHIP.....</b>	<b>19</b>
<b>ANNEX 2 - CONTENT FILE EXAMPLES .....</b>	<b>23</b>

<b>ANNEX 3 – COORDINATES TRANSFORMATION MATRIX .....</b>	<b>26</b>
<b>ANNEX 4 – FROG STATE EDITOR .....</b>	<b>29</b>

## Summary

The goal of this deliverable is to report the implementation details of the Augmented Reality module in the FROG project as well as to provide preliminary results of the performance evaluation. Since the initial conceptualization of the project, Augmented Reality has been considered a distinctive feature that can enhance the attractiveness of a robot guide. While the social aspects of the robot behavior are built upon the analysis of human guides, augmented reality is clearly a content presentation strategy that is unique to computational media. The FROG robot was designed to include hardware and software components that allow and facilitate the presentation of this type of content. It includes a monitor, a video projector, a laser pointer, LED lights and speakers that are directly controlled by this module, supplying services that are used by the Behavior module.

Considering that Mobile robots, such as FROG, and Augmented Reality share a crucial characteristic, localization, this module makes use of the precise 6DOF localization information provided by the navigation module in order to calculate multimedia content localization.

Although the AR module makes use of the localization information from UPO, it was decided that it would have its own coordinates system to achieve a higher independency. The objective of this strategy was to facilitate the decoupled development of the several software modules and so that the AR module can be independent from the robotic framework where it is applied. This could also lead to a software product of its own as a functionality that can be added to other robots, making use of different localization and coordinates systems. This adequacy is easily obtained by establishing a transformation matrix to each system.

The relationship and communication of the AR module with the other modules in the system follows the publish/subscribe model defined in the integration framework.

This module is also responsible for playing the expressive animations of the antenna. A specific tool was developed to design these animations.

## 1. Introduction

The Augmented Reality (AR) Module is responsible for providing services on Augmented Reality and Multimedia. These services are mainly requested by the Behavior Module, but they can also be triggered when a set of predefined conditions are satisfied. These services are the following:

- Show an image
- Show a 3D model
- Play a video
- Play a sound
- Point the laser
- Play antenna animation

The FROG robot provides two different image output devices with which imagery content can be displayed: video projection and monitor. The AR module is also responsible for displaying the content through the adequate output, according to the request.

The module also contains services to retrieve information about space located items from the point of view of the Robot. These services are used by the Behavior Module by request or when a set of conditions are met.

As an example, the robot in an arbitrary position and orientation can use the antenna to point in the direction of the exit or the nearest toilet (this will not be implemented but illustrates an example of the module services potential). The system can also ask the module if some item in real space is within a specific range of directions relative to the robot.



## **2. AR and localized content**

During execution, the module converts references to Locations in Real Space, back and forth between the robot's local point of view at any moment, and an external absolute referential. For AR purposes, some references are projected according to the model of the output device it is destined to use.

The notion of the robot's current position and orientation is obtained from UPOLOC module, in regular updates.

In configuration time, Locations can be defined in Real Space, so they are known in the global system and can be used as route points, points of interest, and locations for content. Content items may be attached to Locations in Real Space, by the use of Smart Objects, which will be explained further on in this document.

### 3. Antenna



Fig. 1 – Antenna/Laser pointer rendering

The FROG robot includes an antenna which is functionally a robotic arm with a laser pointer at its end. This antenna on top of the FROG robot fulfills several objectives. It includes a laser pointer in order to point and highlight specific points in space and it is an expressive vehicle to convey information about the robot internal affective state (much like a dog tail).

#### ***3.1 As a pointing device***

The coordinates of any given Locations in the real world can be computed relative to the robot. Then the antenna can point to that Location using inverse kinematics.

#### ***3.2 As an expression component***

Being an articulated device, the antenna can perform gestures in order to express emotions. Animations can be recorded, which can be read by this module, and used as affective expressive content. These animations can be performed upon request or when a set of predefined conditions are met.

## **4. Integration in the robot framework**

Most of the interaction the AR module has in the robot framework is with the Robot Behavior Module. Apart from the Robot Behavior Module, the relationship with other modules of the system is made through the general integration framework.

### **4.1 The Content Layer**

A layer of protocol logic and conventions was developed, in order to provide a common notation to modules interacting with AR and Multimedia features included in this module.

Over this layer a set of objects can be defined, which are valid through different parts of the framework, and can be used by other modules in the global framework. Two of the most important structures are Smart Objects and Routines.

### **4.2 Smart Objects**

Smart Objects are software objects that can aggregate content (data) and the logic (code) to manipulate it. A smart object contains all the information about itself so that it may be “consumed” by the robot framework. An example can be an object that includes a 3D model with all the logic for its exhibition and, eventually, the possible user interactions.

An important advantage of using Smart Objects, is the fact that a single Player can be developed and used generically, since it is the Smart Object who will be “telling” the player what to do with it. This avoids the development of a different player for each pair of content/behavior type.

Smart Objects benefit from some autonomous execution, which provides a considerable degree of protection from affecting one another in their execution.

Smart Objects can be known and referred to, throughout the framework, by the modules that include the Content Layer. In this case, only the behavior module includes this layer, however, the AR module was built so that any other can include it. An example could be a Smart Object that is played by the navigation module when a specific position is reached or an even lower level in an alarm situation.

For most Smart Objects, the routine is expected to manipulate the Content Objects assigned to them.

### 4.3 Behaviors and Routines

Behaviors are logical units that can execute at the AR Module with some degree of independence. They can be requested to execute or be triggered by the occurrence of a pre-specified event; in this case they are also called Routines. Other Behaviors can be associated with Smart Objects, these Behaviors are triggered by object activation. A set of contextual information is passed into each Behavior execution, this set includes information acquired from outside the module and module internal information. A Behavior may define a set of properties that can be used to pass values into a particular execution.

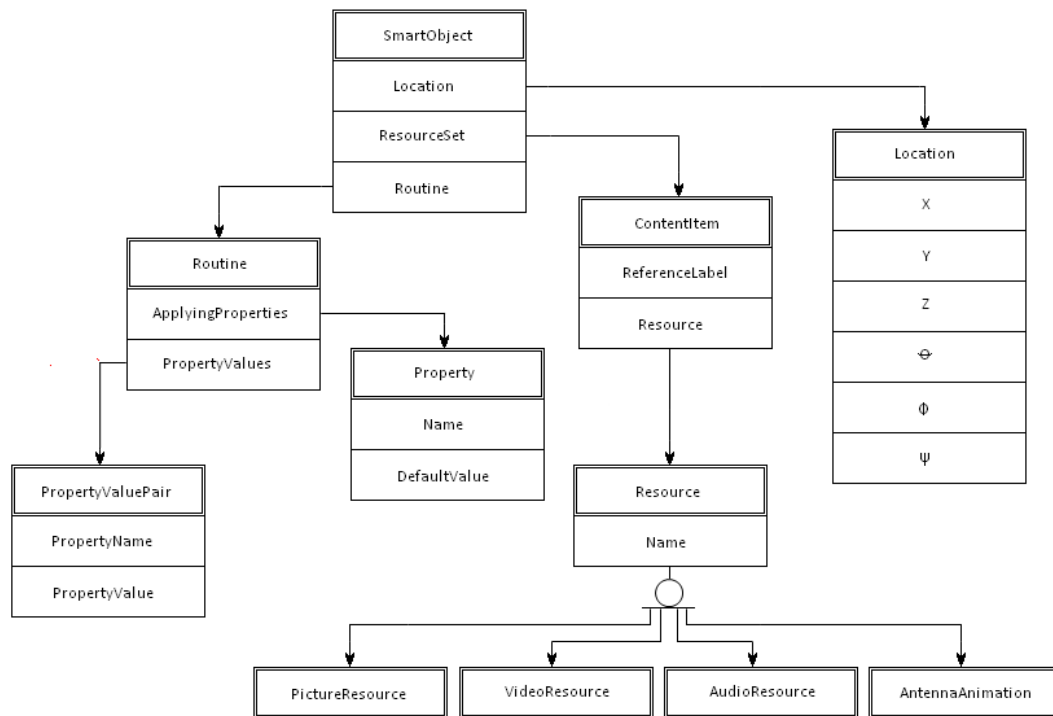
Every Routine execution request has an identifier which should be unique for each requester. The requester should subscribe the *ExecutionStarted*, *ExecutionFinished* and *ExecutionCancelled* topics, so it can be notified of the start, the finishing or the cancellation of the execution requested.

Behaviors associated with Smart Objects are expected to access the content of the object and possibly the location.

Other type of Behaviors, called Questions also define a set of possible answers. The requester subscribes the Answer topic to receive the answer.

## 4.4 Relationships between types

Some of the most important types are represented in the diagram below, along with the relationships between them.



## **5. Managing Content**

The AR module expects specifications and configurations of content in a set of text files in predefined file system locations.

Typically, raw content items, such as Videos, Images and the like are specified for loading. Then more complex structures are defined with reference to content items provided.

The main Media Resource Types we have considered for FROG are: Picture resource, Video resource, Sound resource, Antenna animation Resource.

### **5.1 Content Files**

Raw content items and other objects of the Content Layer are stored in text files in the local file systems.

An example of a Smart Object with a routine and a few content items, is presented in annex 2.

### **5.2 Locations**

In order to assign a location to a Smart Object, the latter has to be configured with the location coordinates.

The coordinates can be acquired with the help of a calibrated plane of the area, as provided by this module. By picking a location on the plane, their coordinates are displayed. If the object is to be placed at a specific height, this value must be added to the coordinates, because only locations at floor level can be picked from the plane.



## 5.3 Content Manager

Content management software was developed to ease the task of specifying configuring and relating content.

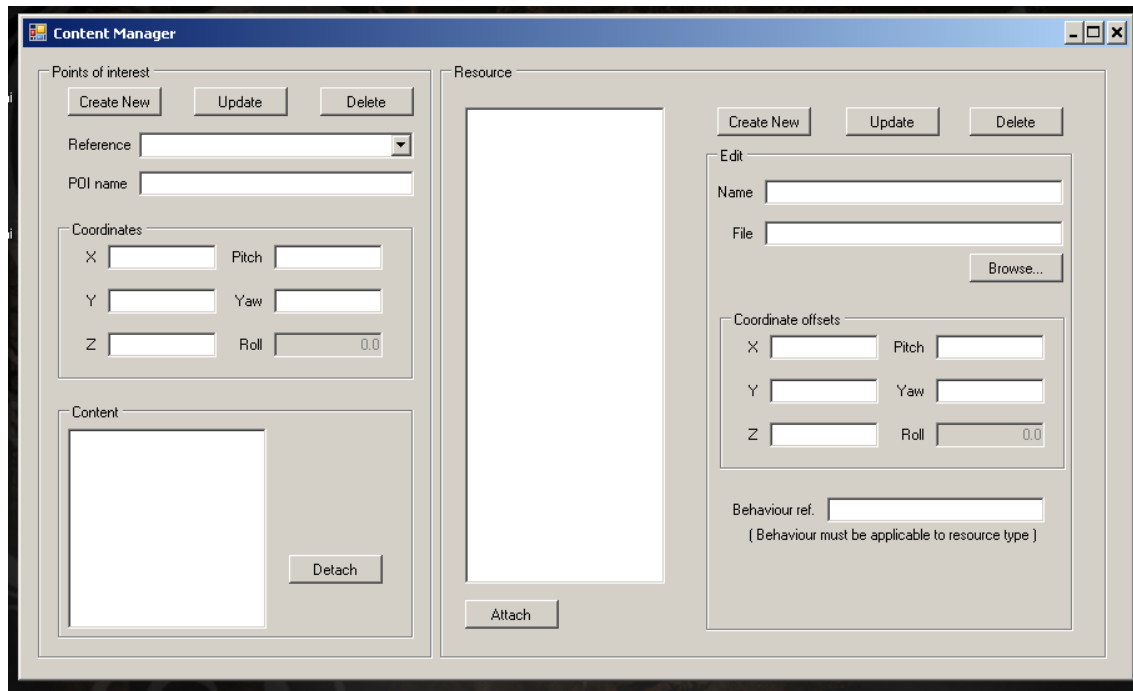


Fig. 2 - backoffice content manager GUI

Using this software, content files can be specified to be loaded into the system, Smart Objects can be created, locations can be given, and routines attached. The software updates the content data files accordingly.

## 6. Particular coordinate space

So that content level configurations, do not get their meanings altered by changes in the navigation system, content locations are defined in a separate coordinate space.

At runtime, locations specified in the global referential, like the position updates from the navigation module, must be converted into this Particular coordinate space before being used. Before leaving the Content Layer, coordinates passed unto the global framework must be converted to the Navigation Referential.

### 6.1 Space Transform

Two real space points were defined, to be used as calibration references. Every time a new scenario is mapped and every time the navigation referential changes, the new coordinates for the calibration reference points must be provided. The new transform can then be calculated.

This transform is determined by a 4x4 matrix. A point can be multiplied by such a matrix, after being converted into homogeneous coordinates and interpreted as a row vector.

Conversion of point into homogeneous coordinates

$$P(p_x, p_y, p_z) \rightarrow P_h(p_x, p_y, p_z, 1)$$

Multiplication by the transform matrix

$$\begin{matrix} p_x & p_y & p_z & 1 \end{matrix} \times \begin{matrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ t_{41} & t_{42} & t_{43} & t_{44} \end{matrix} = \begin{matrix} q_x & q_y & q_z & 1 \end{matrix}$$

The considered referentials have the same scale and their y-axes are parallel. The final transform must include a rotation about the y-axis a translation. A handedness conversion must also be included, since the referentials have opposite handedness.

Annex3 contains the calculations of the transform extraction for a specific case at the Real Alcazar scenario. The extraction process is explained next.

## 6.2 Transform extraction

Based on the coordinates of reference points specified in UPOLOC referential, named Ref1<sub>U</sub> and Ref2<sub>U</sub>, a matrix is calculated at module startup, for converting points from Content space to UPOLOC space, and its inverse for converting back from UPOLOC space to Content space.

U : Referential used by UPOLOC

C : referential used by AR Module (Content space referential)

Ref1<sub>U</sub>, Ref2<sub>U</sub> : reference points in coordinates of U.

Ref1<sub>C</sub>, Ref2<sub>C</sub> : reference points in coordinates of C.

### 6.2.1 Left/Right Handedness

UPOLOC uses a right-handed referential, while the Content space referential is left-handed. The reference point coordinates from U are first mirrored about the xy-plane, considering a left handed referential we will call U<sub>L</sub>, whose y and x-axis coincide with those of U, but the z-axis is directed in the opposite way. All further operations in the transform extraction method are used as defined for left-handed coordinate systems. At last, the handedness conversion will be included into the final transform matrix.

Rotation angles are negated in handedness conversion, in order to be preserved between left and right-handed rotation conventions.

Mirroring a point about the xy-plane is equivalent to negating the z component of the coordinates or applying the handedness conversion matrix:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Calculating the U<sub>L</sub> version of the reference points:

$$\text{Ref1}_{U_L} = \text{Ref1}_U \times H$$

$$\text{Ref2}_{U_L} = \text{Ref2}_U \times H$$

## 6.2.2 Rotation from C to U\_L

By convention, the xz-planes of both referentials are parallel. This means in terms of rotation, that referentials U\_L and C can only differ by yaw.

To determine the angle from C to U\_L the vector from Ref1 to Ref2 is calculated in both versions:

$$\begin{aligned}\vec{V_C} &= \text{Ref2}_C - \text{Ref1}_C \\ \vec{V_{U_L}} &= \text{Ref2}_{U_L} - \text{Ref1}_{U_L}\end{aligned}$$

The angle between  $\vec{V_C}$  and  $\vec{V_{U_L}}$  is then computed. The direction of the rotation is calculated using the cross product vector, which is collinear with the y-axis:

$$\begin{aligned}\vec{cProd} &= \vec{V_C} \times \vec{V_{U_L}} \\ \text{signal} &= \frac{\vec{cProd}.y}{|\vec{cProd}.y|} .\end{aligned}$$

The rotation magnitude is calculated using the dot product between vectors:

$$\text{angle} = \arccos\left(\frac{\vec{V_C} \cdot \vec{V_{U_L}}}{\|\vec{V_C}\| \times \|\vec{V_{U_L}}\|}\right) .$$

The angle rotation around the y-axis,  $\theta$ :

$$\theta = \text{signal} \times \text{angle} .$$

The matrix representing this rotation is a y-axis rotation matrix:

$$M_{\text{rot}} = \text{Rot}_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

### 6.2.3 Translation from C to U\_L

To compute the translation vector from C origin to U\_L origin, one of the reference points is chosen and its coordinates in C referential, are rotated back to align with U\_L referential.

$$\text{Ref1}_{\text{CRot}} = \text{Ref1}_C \times M_{\text{rot}} \quad .$$

T translation vector will be:

$$\vec{T} = \text{Ref1}_{\text{U}_L} - \text{Ref1}_{\text{CRot}}$$

and the translation matrix will be:

$$M_{\text{trl}} = \begin{matrix} & \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{matrix} \\ \begin{matrix} \vec{T}_x \\ \vec{T}_y \\ \vec{T}_z \end{matrix} & \begin{matrix} \vec{T}_x \\ \vec{T}_y \\ \vec{T}_z \end{matrix} & \begin{matrix} \vec{T}_z \\ 1 \end{matrix} \end{matrix} \quad .$$

### 6.2.4 The transform matrix

The transform matrix from C to U\_L can now be calculated:

$$M_{\text{C2U}_L} = M_{\text{rot}} \times M_{\text{trl}} \quad .$$

The final transform matrix transforms from referential C to referential U, so it has to include the handedness transform.

$$M_{\text{C2U}} = M_{\text{C2U}_L} \times H \quad .$$

For converting back from U to C the inverse of  $M_{\text{C2U}}$  is used:

$$M_{\text{U2C}} = M_{\text{C2U}}^{-1} \quad .$$

## **7. Evaluation of the AR application performance and the effectiveness of the interaction strategies**

In order to evaluate the AR application performance and the effectiveness of the interaction strategies a field test was held during the integration and data collection week in the Lisbon Zoo in July 2013.

In order to prepare this field test, the table with multimedia content of Deliverable 2.3 was used and complemented with suggestions for robot interaction strategies that resulted from research in WP4. Two storyboards were worked out in more detail for the test. The first storyboard used a wall projection in the Tigers' Valley and the second storyboard used the antenna to point with the laser at an information panel in the Primates' Temple. The test was a Wizard of Oz test hence the movements of the robot and the sound were manually operated.

Unfortunately, shortly before the field test started it appeared that the speakers of the robot did not work. There was not enough time to find the cause of this defect (which later appeared to be quite easily solvable). A laptop was used for the sound instead. Consequently the volume of the sound was far too low and the robot could not be heard, neither by the visitors of the zoo, nor by the person operating the robot, so the field-test mainly failed.

The most important thing we learned from this test is that the zoo is a very noisy environment. At most places the background noise from animals, birds, waterfalls and people is very loud. Therefore it is recommended to use powerful speakers that can be directed to the visitors that are being addressed by the robot.

Later we learned that the robot's speakers would not have succeeded in overcoming the noise either. At the time of writing IDMind is looking for more powerful speakers and YDreams is adapting their presentations to ambient noise levels at both sites.



## Annex 1 - Types in AR and in AR-Behavior relationship

- **question** (questionId) - Predefined routine, to which a user-chosen answer is expected.

- **answer** (answerId) - Predefined answer, which can be one of the expected answers for each defined question.

- **resource** (resId) - Playable on one or more displays depending on resource type. Can be associated to a behavior and a location with an object.

### **resource types:**

.**sound** - A sound or sequence of sounds. These resources may represent noises, speech tokens or music.

.**graphical object** - 3D model, 2D object with some content (video, text, etc.) or other visual effect.

.**antenna animation** - predefined sequence of commands to FROG antenna in direct or inverse kinematics fashion.

- **object** (old) - Object with a location in the world. May contain one or more playable resources, and a behavior. The behavior may reference to the resources and the location as well as contextual data.

- **display** (displayId) - May be one of { Screen, Projector1, AllValidDisplays, Antenna, Speaker 1, Speaker 2, Speakers }

- **logicalPredicate** (predicateId) – Predefined or derivative predicate whose value may depend totally or partially on module inputs.

- timeElapsed (timeInterval);

- momentReached (dateTime)

- inRangeOf (location, radius)

- isPeopleDetected

- and (predicateId, predicateId)

- or (predicateId, predicateId)

- not (predicateId)

derivative predicates may be composed based on predefined predicates. Predicates are provided to behaviors as part of the context.

- **behavior** (bId) - Basic or derivative behavior, applicable to a specific domain of resource types.

basic behaviors:

    \_ display (parameters: duration, repeat) (applies to: sound, graphical, antennaAnim)

- displayAt (parameters: same as play plus set of displayId's) (applies to: sound, graphical)

- stop/turnOff (no parameters) (applies to: sound, graphical, antennaAnim)

- changeLocation/pointTo (parameters: worldCoordinateSet or predefinedPlace) (applies to: sound, graphical, antennaAnim (make antenna point at location))

- onCondition (parameters: predicateId);
- sequence (params: beh\_1, ..., beh\_N, failPolicy)
- parallel (params: beh\_1, ..., beh\_N, failPolicy)

derivative Behaviors : constructs based on other behaviors, suitable for more complex procedures.

default behavior – predefined behavior, executes at module start.

- **behaviorProperty** (propId) – placeholders where to pass parameters to and from behaviors invoked.

Behaviors define a set of applicable Properties and default values for each one. If one of the properties is not mentioned at execution start, the default value for that property will be used.

Properties:

Duration (time)

Location (old or coordinates)

ScreenPosition (2D coordinates)

Question (questionId)

Answer (answerId)

Object (old)

Display (displayId)

StartPolicy (startPolicy)

ExecId (execId)

...

- **execId** - Denotes a particular execution of a behavior
- **executionResult** (resultId) - One of { finished, canceled }
- **startPolicy** - One of { Now Next, DontCare }

## Module Inputs/Outputs (subscribe/publish)

- [In] Start  
Same as Execute[DefaultBehavior { StartPolicy = Now | Dontcare; ExecId = null}]
- [In] Stop  
Same as: Cancel[ ExecId = null]
- [In] Execute [behId, startPolicy, execId, {Property0 = val0, Property1 = val1, ...} ]
- [Out] Execution Started [ExecId = execId]

- [Out] ExecutionFinished [{ ExecId = execId; Result = resultId}]
- [In] LocationUpdate [coordinates]
- [In] PeopleTracked [{peopleTracked = [(id0, cords, gesture), (id1, cords, gesture), ...] }]
- [In] Cancel [ExecId = execId]
- [In] Question [QuestionId = questionId]
- [Out] Answer [QuestionId = questionId, Result = resultId]

## Annex 2 - Content file examples

---

Resource File: "Tilemap.res"

---

```
{
    "Name" : "Tilemap",
    "Filename" : "tilemap04.jpg",
    "CoordinateOffsets" :
    {
        "X" : 0.0,
        "Y" : 1.233,
        "Z" : 0.0,

        "Pitch" : 0.0,
        "Yaw" : 0.0,
        "Roll" : 90.0,
    },
    "TimeOffset" : 0.0
}
```

---

ResourceFile: "TileSong.res"

---

```
{
    "Name" : "TileSong",
    "Filename" : "tilemusic.mp3",
    "CoordinateOffsets" :
    {
        "X" : 0.0,
        "Y" : 0.0,
        "Z" : 0.0,
        "Pitch" : 0.0,
        "Yaw" : 0.0,
        "Roll" : 0.0,
    }
}
```

```

    },
    "TimeOffset" : 4.0
}

```

---

Routine File: "ProjectTilesWhilePlayingMusic.routine"

---

```

{
    "Name" : "ProjectTilesWhilePlayingMusic",
    "Reference" : "ProjectTilesWhilePlayingMusic",
    "ApplicableProperties" :
    [],
    "ContentProperties" :
    [
        {
            "Name" : "Tilemap_IMAGE"
        },
        {
            "Name" : "TileMusic_AUDIO"
        },
        {
            "Name" : "TileAnimation_VIDEO"
        }
    ],
}

```

---

Smart Object File: "TapestryRoomWall.sobj"

---

```

{
    "Name" : "TapestryRoomWall",
    "Reference" : "TapestryRoomWall",
    "CoordinateOffsets" :
    {
        "X" : -225.452,
        "Y" : 0.0,
    }
}

```



```

        "Z" : -183.038,
        "Pitch" : 0.0,
        "Yaw" : 0.0,
        "Roll" : 0.0,
    },
    "BehaviourReference" : "ProjectTilesWhilePlayingMusic",
    "ContentPropertyValues" :
    [
        {
            "PropertyName" : "TileMap_IMAGE",
            "Value" : "Tilemap04.jpg"
        },
        {
            "PropertyName" : "Tilemusic_AUDIO",
            "Value" : "TileSong.mp3"
        }
    ]
}

```

## Annex 3 – Coordinates Transformation matrix

The example here refers to the Real Alcazar scenario. The reference points chosen are signaled in the following picture:



Fig. 3 – Coordinates spaces referential at the Real Alcazar

The coordinates of the chosen points:

In C referential:

Ref1<sub>C</sub> (-73.24401, 0.19, 57.36475)

Ref2<sub>C</sub> (-66.32367, 0.19, 65.12097)

In U referential:

Ref1<sub>U</sub> (107.2, 0.19, 93.40)

Ref2<sub>U</sub> (108.6, 0.19, 83.10)

Converting coordinates from U to U<sub>L</sub>

Ref1<sub>U<sub>L</sub></sub> = Ref1<sub>U</sub> × H = (107.2, 0.19, -93.40)

Ref2<sub>U<sub>L</sub></sub> = Ref2<sub>U</sub> × H = (108.6, 0.19, -83.10)

Calculating  $\vec{V}$  vectors for both C and U<sub>L</sub> referentials

$$\begin{aligned}\vec{V}_C &= \text{Ref2}_C - \text{Ref1}_C = \vec{V}_{(6.920341, 0.0, 7.756218)} \\ \vec{V}_{U\_L} &= \text{Ref2}_{U\_L} - \text{Ref1}_{U\_L} = \vec{V}_{(1.400002, 0.0, 10.3)}\end{aligned}$$

Extracting the rotation

$$\vec{cProd} = \vec{V}_C \times \vec{V}_{U\_L} = \vec{V}_{(0.0, -60.42082, 0.0)}$$

$$\text{signal} = \frac{\vec{cProd}.y}{|\vec{cProd}.y|} = \frac{-60.42082}{60.42082} = -1$$

$$\text{angle} = \frac{\vec{V}_C \cdot \vec{V}_{U\_L}}{||\vec{V}_C|| \times ||\vec{V}_{U\_L}||} = -0.593411863 \quad (\cong 34^\circ)$$

$$\theta = \text{signal} \times \text{angle} = 0.593411863 \quad (\cong 34^\circ \text{ clockwise})$$

$$M_{\text{rot}} = \begin{pmatrix} 0.8290376 & 0 & 0.5591928 & 0 \\ 0 & 1 & 0 & 0 \\ -0.5591928 & 0 & 0.8290376 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Extracting the translation

$$\text{Ref1}_{C\text{Rot}} = \text{Ref1}_C \times M_{\text{rot}} = \vec{V}_{(-92.8, 0.19, 6.600011)}$$

$$\begin{aligned}\vec{T} &= \text{Ref1}_{U\_L}(107.2, 0.19, -93.40) - \text{Ref1}_{C\text{Rot}}(-92.8, 0.19, 6.600011) \\ &= (200.0, 0.0, -100.0)\end{aligned}$$

$$M_{\text{Trl}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 200.0 & 0 & -100.0 & 1 \end{pmatrix}$$

Transform matrix for converting coordinates from C to U

$$M_{C2U} = M_{\text{rot}} \times M_{\text{Trl}} \times H = \begin{bmatrix} 0.8290376 & 0 & -0.5591928 & 0 \\ 0 & 1 & 0 & 0 \\ -0.5591928 & 0 & -0.8290376 & 0 \\ 200.0 & 0 & 100.0 & 1 \end{bmatrix}$$

Transform matrix for converting coordinates from U to C

$$M_{U2C} = M_{C2U}^{-1} = \begin{bmatrix} 0.8290376 & 0 & -0.5591928 & 0 \\ 0 & 1 & 0 & 0 \\ -0.5591928 & 0 & -0.8290376 & 0 \\ -109.8882 & 0 & 194.7423 & 1 \end{bmatrix}$$

## Annex 4 – Frog State Editor

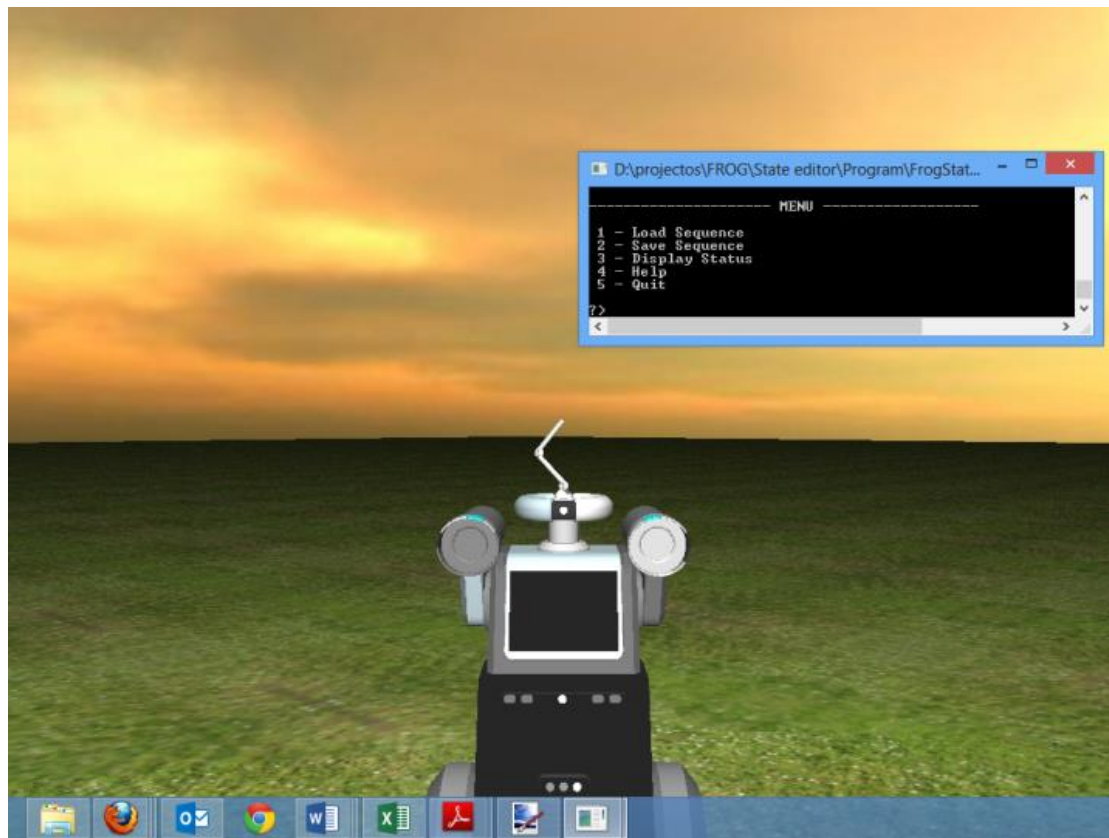


Fig. 4 – FROG state editor

### Features in Frog State Editor :

- viewing the robot model by rotating around it approaching or moving away from it;
- changing state by moving the antenna and changing the screen state;
- construct a **state** sequence by adding and removing state;
- play the whole current sequence or play just one step and go back one step;
- save sequences to files, load sequences from files

### Windows

Frog State Editor uses two windows:

- a graphics window, which normally starts on top and where the robot is displayed;
- a text console where some more general commands can be given.

## **Graphics Window**

### **Moving the camera**

In the Graphics window the camera view can rotate around the robot by keeping the right mouse button down while moving the mouse.

The view can get near or move away from the robot by turning the mouse wheel.

### **Changing the robot state**

The three joints of the antenna can be actuated:

Base Joint – keys ‘z’ and ‘x’;  
Middle Joint – keys ‘q’ and ‘a’;  
Top Joint – keys ‘w’ and ‘s’.

The screen state can be changed:~

Screen off – key ‘1’;  
Screen displaying video – key ‘2’;  
Screen displaying smile – key ‘3’.

### **Editing the sequence**

Initially, there’s just an empty sequence. A sequence can be loaded from file or built up by adding states. Current state of the robot can be added to current sequence by pressing ‘m’ key and last state can be removed from sequence by pressing ‘n’.

To play the current sequence press ‘p’. While sequence is playing no state can be added or removed nor can the sequence be stepped.

Initially, the current state is the first in the sequence. The transition to next state can be played or Current can be moved back to previous, as long as these positions exist.

To Step forward press ‘right arrow’ key , to go back one step press ‘left arrow’ key.

## Console Window

A menu is displayed in the console Window. An option from the menu can be chosen by entering the corresponding digit and pressing Enter.

The following options are on the menu:

- 1 – Load sequence from file. When this option is chosen, the user is asked for a filename to load. The file must exist at the 'Sequences' folder, within the program folder.
- 2 – Save sequence to file. The user is asked to provide a name for the file. A file with that name must not exist. Note that in the case a sequence is loaded, changed and saves again it must be saved to different name or the original file must be renamed or removed.
- 3 – Display status, displays info about the current sequence editing : filename, number of states in the sequence and the current state.
- 4 – Help. Displays key functions.
- 5- Quits the application.