SAPHARI

SAFE AND AUTONOMOUS PHYSICAL HUMAN-AWARE ROBOT INTERACTION

Project funded by the European Community's 7th Framework Programme (FP7-ICT-2011-7)
Grant Agreement ICT–287513

Deliverable D5.1.1

Gesture and grammar descriptors of human motion and statistical gesture parser

Milsetone due date: 30 April 2015	**Actual submission date:**
Start date of project: 1 November 2011	**Duration:**
Lead beneficiary for this deliverable: UNINA	**Revision: 0.#/1.#/Final**

Nature: R	**Dissemination Level: CO**

R = Report
P = Prototype
D = Demonstrator
O = Other
	PU = Public
	PP = Restricted to other programme participants (including the Commission Services)
	RE = Restricted to a group specified by the consortium (including the Commission Services)
	CO = Confidential, only for members of the consortium (including the Commission Services)

www.saphari.eu

Executive summary

This deliverable describes gesture recognition and semantic interpretation methods in the context of the multimodal human-robot interaction (HRI) framework described in MS22@24. We illustrate the overall interaction architecture and the associated algorithms and methods for gesture classification, multimodal fusion and semantic interpretation of the human intentions (see also MS36@36). Moreover, we illustrate two instances of the proposed framework at work in the AIRBUS and in the DLR Hospital domain respectively.

The document is structured as follows: in the first section, the multimodal architecture is presented describing each involved module: the gesture classifier, based on Latent Dynamic Conditional Random Fields (LDCRFs), the speech classifier, based on the Julius speech recognition engine, and the multimodal fusion engine that provides the semantic interpretation of the overall classified results exploiting stochastic grammars. In the second section, we detail specific instances of the multimodal interaction framework focusing on two SAPHARI case studies: the AIRBUS domain and the DLR Hospital domain. Finally, in these scenarios, we describe the overall framework at work during cooperative task execution.

Publications

The Following articles concerning the contents of this milestone have been published:

1. **Interacting with robots via speech and gestures, an integrated architecture**, F. Cutugno, A. Finzi, M. Fiore, E. Leone, S. Rossi. *INTERSPEECH 2013 - 14th Annual Conference of the International Speech Communication Association*, Lyon (France), August 2013.
2. **Multimodal Interaction and Attentional Regulation in HRI**, F. Cutugno, A. Finzi, S. Iengo, S. Rossi and M. Staffa. *6th International Workshop on Human-Friendly Robotics (HFR 2013)*, Rome (Italy), October 2013.
3. **An Extensible Architecture for Robust Multimodal Human-Robot Communication**, S. Rossi, E. Leone, M. Fiore, A. Finzi and F. Cutugno. *IEEE International Conference on Intelligent Robots and Systems - IROS*, Tokyo (Japan), November 2013.
4. **A Dialogue System for Multimodal Human-Robot Interaction**, L. Lucignano, F. Cutugno, S. Rossi and A. Finzi. *15th International Conference on Multimodal Interaction – ICMI2013*, Sydney (Australia), December 2013.
5. **Continuous Gesture Recognition fo Flexible Human-Robot Interaction**, S. Iengo, S. Rossi, M. Staffa and A. Finzi, *in Proceedings of IEEE International Conference on Robotics and Automation - ICRA 2014, May 31- June 7, 2014. Hong Kong, China*.

Table of contents

Architecture for Multimodal Human-Robot Interaction

In milestone MS22@24 we described the Multimodal Human Robot Interaction (MHRI) architecture proposed for the SAPHARI domain. The proposed architecture is structured in different levels (see Figure 1) and it is based on a late fusion strategy, where the recognition is delayed after the recognition of single modalities. Details about the architecture can be found in [Cutugno13, Rossi13].

In the following, we briefly illustrate its main components.

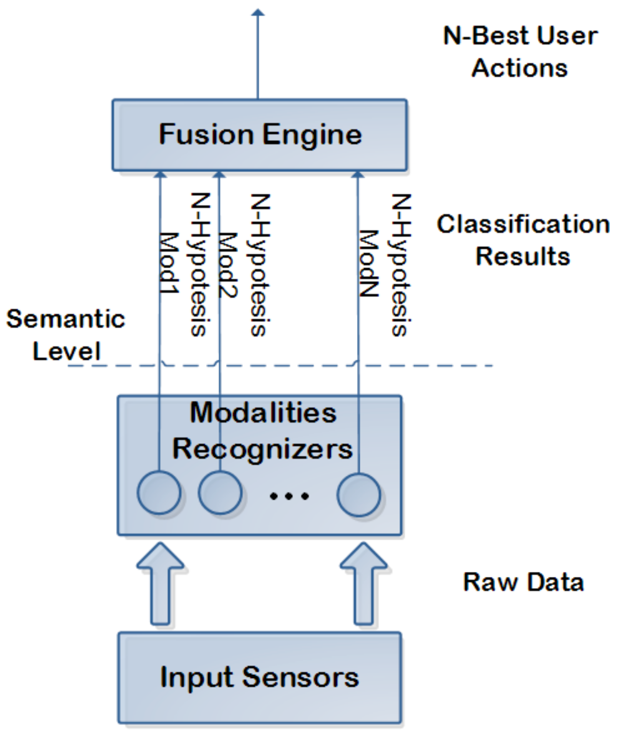


Figure 1: MHRI Architecture.

- **Input Sensors**: in this module all the external data are collected and then stored in feature vectors. We considered two main interaction modalities: gestures and speech, however, other modalities can be easily integrated.
- **Modalities Recognizers**: in this module a separated classifier for each modality is provided. As a result of this process, each features vector is associated to a list of N-best probabilities with respect to the possible commands. In particular, we considered the following user commands: Point at, Take, Give, No, Find, Come, Leave, Stop.
- **Fusion Engine**: this module represents the core of the system (see Figure 2), since it is responsible of fusing the different choices made in previous modules and eventually of re-interpreting them through a late fusion approach. In the following, we will briefly illustrate the usage of probabilistic context-free grammars, which provide the N-best actions list for multimodal interpretation process.

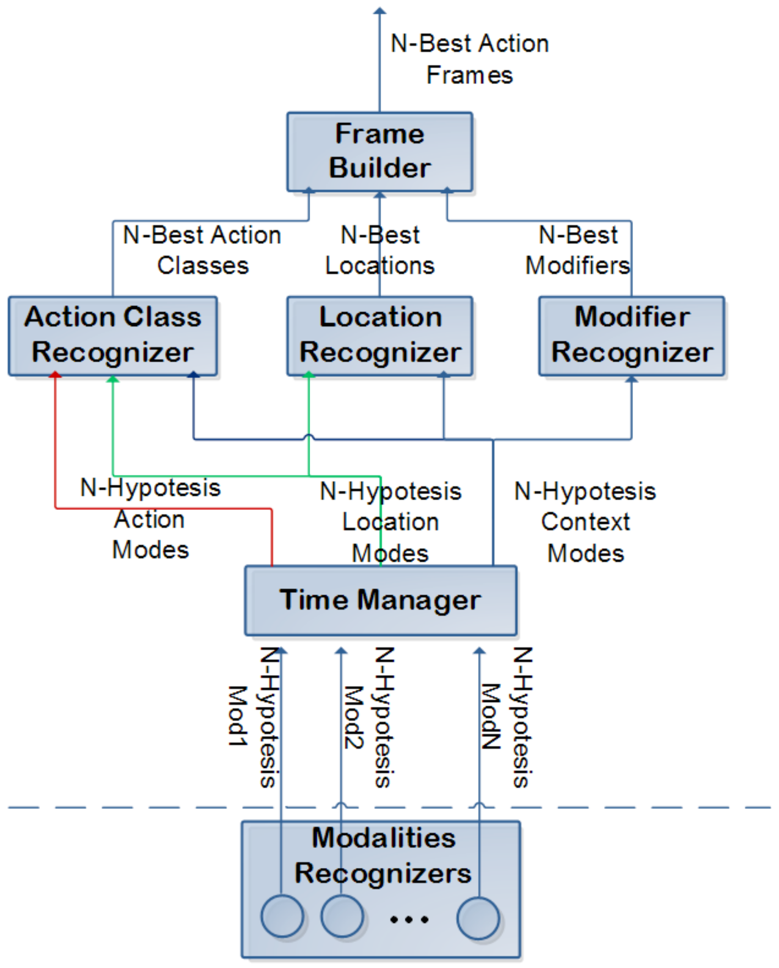


Figure 2: Fusion Engine.

The Fusion Engine is composed of the following components:

Time Manager: this module is to synchronize the inputs coming from the considered interaction modalities. This synchronization process allows the fusion engine to group classification results belonging to the same time interval. In particular, given a suitable temporal threshold, it discriminates whether interpreted commands from different channels are parallel or consecutive.

Action Recognizer: this module is to assess the N-best list of human intentions given the synchronized results of single classifiers. Different techniques can be deployed for this purpose. We, here, rely on Probabilistic Context Free Grammar (PCFG) technique.

Location Recognizer: this module extracts spatial information from the executed action. For example, it extracts targets for the specific action.

Mode Recognizer: this component is to interpret the way in which the action is executed by exploiting contextual information.

Frame Builder: This module integrates the results of the other modules into one instantiated intention. This phase is also exploited to complete partially specified commands (e.g., a commands lacking some arguments like a location or an object). Specifically, if different received commands are incomplete, but compatible with each other, it tries to fuse them into one single and complete command, otherwise these are left as different commands.

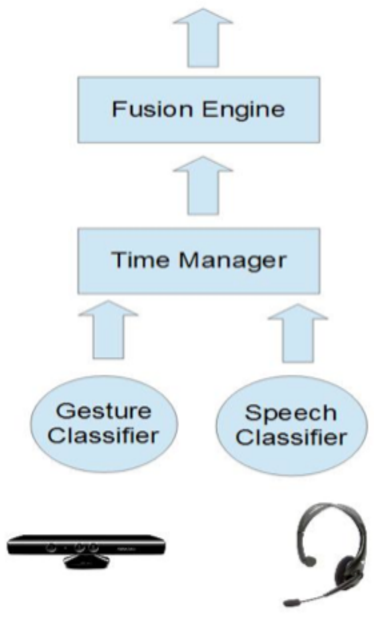


Figure 3: Multimodal fusion engine module integrating gestures and speech.

Implementation

In our setting, we assume the system endowed with a RGB-D camera (e.g., Microsoft Kinect) for gesture recognition and a microphone for speech recognition (see Figure 3). In the proposed architecture the low level is composed of the gesture and the speech package. Each of them provides the classification results to the higher level represented by the fusion package (see Figure 4). We adopted the Robotic Operating System (ROS) for the implementation.

Fusion package. The developed fusion engine module exploits: (i) the Time Manager for synchronization, (ii) the stochastic grammar technique for the late fusion and (iii) a frame builder for collecting final results. The Time Manager synchronizes the classified data separately generated by the interaction channels: each classified result is associated to a temporal interval, while temporal constraints (represented by interval overlap relations [Allen83]) are used to decide whether the results obtained from different classifiers should be grouped and interpreted together. Once multiple classified results have been synchronized, the stochastic grammar is used to merge them providing a N-best list of possible interpretations of the overall human intention.

Figure 4 illustrates the overall architecture along with the main involved modules (ROS nodes).

In the following, we will provide a more detailed description of each module.

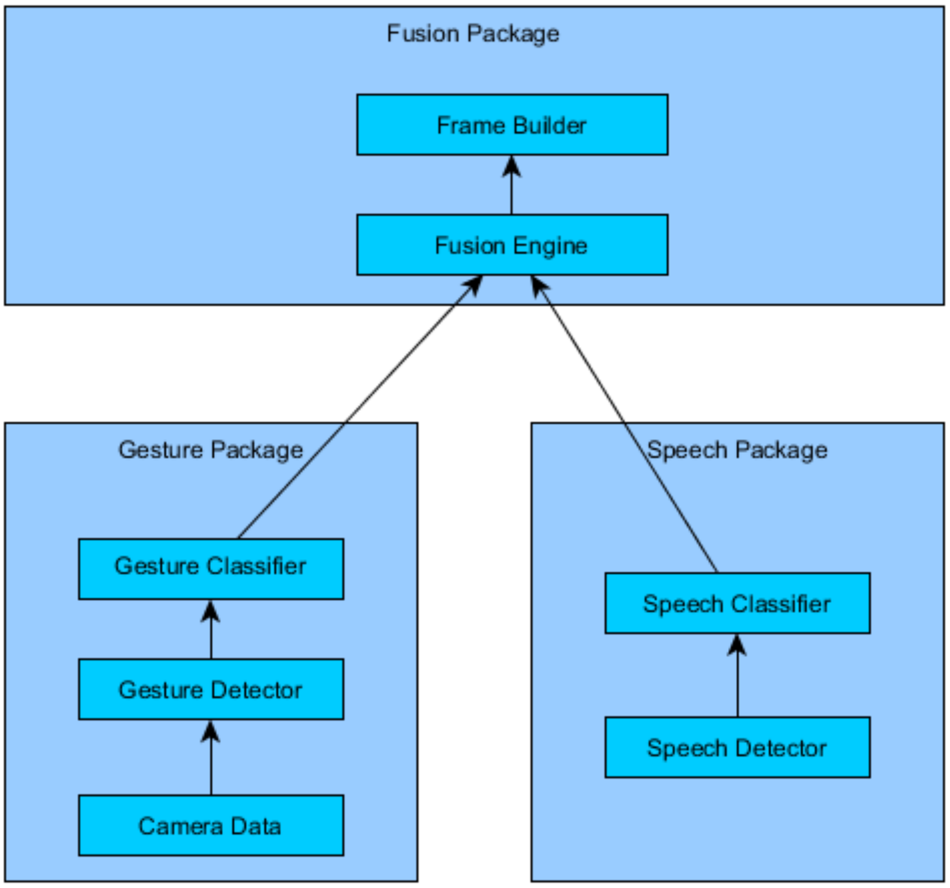


Figure 4: MHRI ROS-nodes architecture.

Gestures package. The Gesture Package is composed of three modules: Camera Data, Gesture Detector and Gesture Classifier. The first module catches the Kinect data sending it to the Gesture Detector, which collects the features needed for the classification process. These features are calculated starting from positions of user joints and the angles between them. More precisely, the Gesture Detector node calculates the hand and elbow joints 3D positions normalized with respect to the shoulder position, the 3D angles of elbow and hand joints, and the hand pose. Initially, the hand pose was obtained with the support of a colored glove allowing us to discriminate among closed, open, and a pointing hand [MS22@24]. The implementation has been, here, improved for permitting the Gesture recognition working without the need of the glove. Namely, the hand identification is obtained by detecting the fingers by using the K-Curvature algorithm and searching for major convexity defects [Rosenfeld73]. The hand image is first isolated exploiting the depth map. Then, the new image is processed with a sharp filter that allows to extract the contours of the image. The K-Curvature detection is applied to the contour points in order to find a curvature from the angle compound by two vectors. Once a contour point is selected, the forward and backward vectors are computed, respectively, considering the k forward points and k backward points.

More formally, if $P = \{p_1, \dots, p_n\}$ is the set of contour points, for each point p_i it is possible to define:

- a forward vector: $f_{ik} = p_i - p_{\{i+k\}}$
- and a backward vector $b_{ik} = p_i - p_{\{i-k\}}$

Where $1 \le i \le n$ and k is a constant value.

Now, if the angle between the forward and backward vector is lower than a threshold, the selected point could be in a curve and represent a possible finger in a hand contour. Relevant points are those in a concave space not directed towards the hand center (see Figure 5). The features obtained from hand recognition are: (i) the *hand pose*, evaluated according to the number of detected fingers, (ii) the *bounding box* of the hand, which is proportional to the depth value of hand joint, and (iii) the *fingers direc*tion.

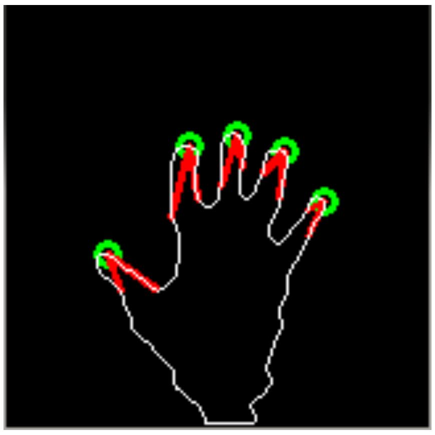


Figure 5: Detected fingers in a hand contour.

The Gesture Detector collects a sequence of frames $\{x_1, \dots, x_n\}$, while the Gesture Classifier labels each frame x_i with a class Y. Hence, the result is a sequence of labels $\{y_1, \dots, y_n\}$. The model used to classify gestures is LDCRF, as presented in M36@36.

LDCRFs [Morency07] are discriminative models, hence better suited for the analysis of continuous data, which cannot be limited by the assumption of observations independence. Indeed, the independence assumption does not suitably capture contextual information and dependencies from distant states. The LDCRF model allows the deployment of hidden variables between observations and classes of objects. Moreover, LDCRFs represent both extrinsic dynamics and intrinsic sub-structures: extrinsic dynamics are learned from a continuous stream of class labels; while, the intrinsic sub-structures are learned by exploiting the intermediate hidden states [Morency07]. Differently from [Morency07], where LDCRFs are used on un-segmented data, but the training and test video sequences are associated with start/end times, in our case we assume that the system is not provided with the gesture start/end times during the recognition phase. Indeed, the Gesture Detector collects the frames in blocks and sends them to the classifier. For each block, the classifier labels each individual frame and builds a list of pairs $(c_y, \#\{f \in c_y\})$, where c_y is the class name and $\#\{f \in c_y\}$ counts the number of labelled frames belonging to the c_y class. The latter number allows us to estimate the probability of the c_y gesture for that block. In order to reduce the noise produced by the real-time gesture recognition, after N blocks of frames, we evaluate the matches of the current results of the gesture recognition with respect to the class of gesture previously recognized (see Figure 6). This way, a gesture is recognized if the associated probability is greater than a suitable threshold after N blocks. For example, in our tests, we consider a gesture recognized if it is classified with a probability value that is greater than 60% for 3 blocks of frames.

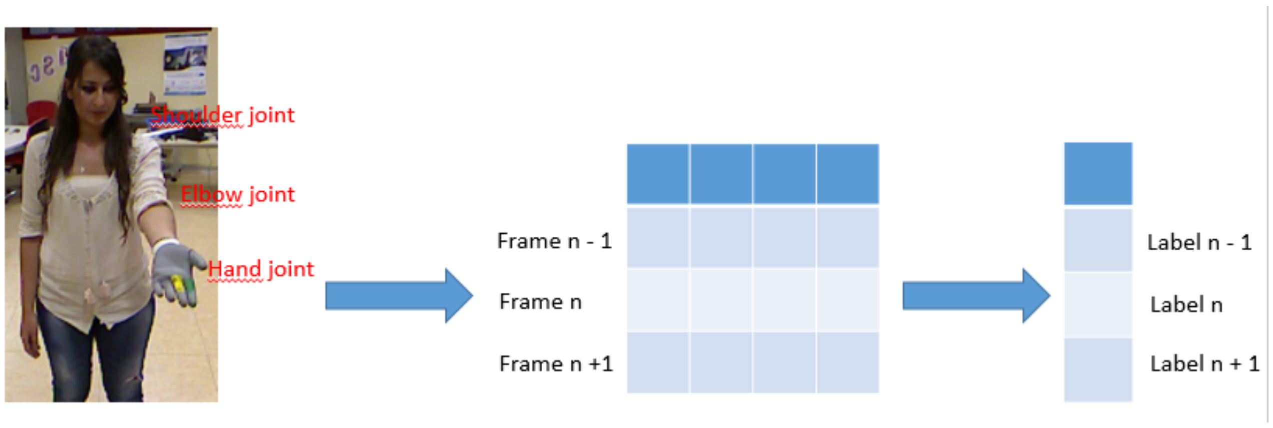


Figure 6: Labelling Frames.

Speech package. The Speech package is composed of the speech recognition and the speech segmentation modules. The speech recognition module exploits the Julius engine [Lee09] for the classification process. The classification result is then provided to the speech segmentation module. Julius is a open-source software for continuous speech recognition with high performance. It is based on N-gram language models and Hidden Markov Model (HMM). The speech is processed in two steps:

- In the first step, the speech is processed using a 2-grams model and a beam search algorithm (high speed, and an approximate search).
- In the second step, a 3-grams model is deployed with a stack decoding search algorithm (high precision). The output is a N-best list of results.

Julius is provided with a grammar and it is trained to recognize a fixed set of combinations of words. Each word of the grammar is coupled with a list of phonemes. The usage of a grammar is feasible because the list of commands to be recognized is not huge. In order to obtain a better result in term of robustness, a keyword, e.g. “robot”, can be used to start the recognition phrase reducing the noise during the continuous speech recognition. Finally, a speech segmentation module generates the command to be sent to the fusion engine. Given the string received from speech recognition module, it generates the triples (l, o, a) representing locations, objects, and action, tagged with the couple (t_s, t_e) representing the time of start/end of the speech.

Fusion package. The fusion engine combines the classification results from the different modalities providing a unique interpretation. The fusion engine receives messages from the Gesture and Speech modules: from the Gesture module, it gets a N-best list of gestures with the associated probabilities and the end time; from the Speech module, it gets the N-best list of recognized words represented as a triple (l, o, a) of locations, objects, and actions tagged with the start/end times (t_s, t_e). When one of the two messages (gesture or speech) arrives to the fusion engine, the latter waits for a certain amount of time F_t in order to intercept a message from the other channel. If no message comes during the waiting time, the fusion is made with the current available information. A stochastic grammar is used to fuse the synchronized classified results. This approach presents several advantages. Indeed, beyond noise filtering, this method enables the representation of an explicit negation of possible interpretations in the fusion process (i.e., when no rule is available for the incoming results of the single classifiers). Moreover, a grammar description of the rules is more readable, easy to be inspected and managed by a human operator when compared with respect to a fusion engine directly obtained from a statistical classifier (e.g. SVM). Following this approach, a stochastic grammar can be exploited in order to recognize the user's intentions as a late modality fusion mechanism. Indeed, the terminal symbols of the grammar can be interpreted as user's intentions (e.g. take an object, stop an operation, etc.), which are derived from rules applied to the results of the single classifiers. In the following, we introduce the Probabilistic Context Free Grammars (PCFG).

Probabilistic Context Free Grammar (PCFG). A context free grammar is defined by the quadruple [Manning99] $G = (N, \Sigma, \mathrm{R}, \mathrm{S})$, where:

- N is a finite set of non-terminal symbols;
- Σ is a finite set of terminal symbols;
- R is a set of rules composed by a symbol in N that it is derived in others symbols in the union set of N and Σ;
- S is the start symbol of a derivation.

The grammar captures a language whose terminal symbols are words. A phrase (compositions of words) belongs to the grammar if there exists a derivation for that phrase. A grammar is ambiguous if there is more than one derivation tree for the same phrase. A probabilistic grammar adds probability values to the derivation rules. Hence, the probabilistic grammar can be defined by a quintuple $G = (N, \Sigma, \mathrm{R}, \mathrm{S}, \mathrm{P})$, where P is the set of probabilities $q(A \rightarrow \mathrm{B})$ for the probabilistic rules belonging to R. Moreover, for each X belonging to N, the following constraint holds:

$$\sum_{A \rightarrow B \in R:\, A = X} q(A \rightarrow B) = 1$$

with $q(A \rightarrow \mathrm{B}) \geq 0, \forall \mathrm{A} \rightarrow \mathrm{B} \in R$.

Let T_G be the set of all derivation trees belonging to the grammar, given a derivation tree $t \in T_G$ that applies the rules $A_1 \rightarrow B_1$, $A_2 \rightarrow B_2$, ..., $A_n \rightarrow B_n$, have that the probability for the tree t is the following:

$$p(t) = \prod_{i=1}^{n} q(A_i \rightarrow B_i)$$

PCFG for late modality fusion. A PCFG can be exploited in order to recognize the user's intentions as a late modality fusion mechanism. Indeed, the terminal symbols of the grammar can be interpreted as user's intentions (e.g. take an object, stop an operation, etc.), which are derived from rules applied to the results of the single classifiers. Our goal is to build a grammar that is suitable for the late fusion classification starting from a training set $S_f = \{(M_{11}, \dots, M_{1n}, A_1), \dots, (M_{m1}, \dots, M_{mn}, A_k)\}$ which is composed of the list of classification results $M_{i1}, \dots, M_{in}$ associated with the expected intention A_j.

The generation process works as follows. We assume the grammar in the Chomsky Normal Form (CNF) and an initial grammar G_0 where:

1) Terminal symbols are the single-modality classifiers results with probability 1 (e.g. grammar words in Table 1);

2) The user intention rules represent the expected fusion results (e.g. T, H, L rules in Table 1);

3) The starting roles are equiprobable (e.g. S-rules in Table 1).

Starting from the G_0 grammar, we generate the fusion grammar G_f by exploiting the training set S_f. This training set is obtained by asking the testers to execute an action or a command in the preferred modalities, and then declaring the actual intention. For each element $(M_1, \dots, M_n, A)$ in S_f , we extract a rule $R: A \ \rightarrow M_1, \dots, M_n$. The probability associated with this rule is given by the number of occurrences of the intention A performed in the modalities $M_1, \dots, M_n$ with respect to the numbers of derivation trees for that intention, that is:

$$q(R) = \frac{\#\{(M_1, \dots, M_n, A) \in S_f\}}{\#\{T_A\}}$$

In the example shown in Table 1, a grammar was trained on three occurrences of intentions: “take”, “leave” and “stop”.

Rules	Score
S → T O	0.25
S → T E	0.25
S → L E	0.25
S → H E	0.25
T → GT ST	0.33
T → E ST	0.33
T → GT E	0.33
L →GL SL	0.33
L → GL E	0.33
L → E GS	0.33
H → GH SH	0.33
H →GH E	0.33
H → E SH	0.33
GT → gesture(take)	1.0
GL → gesture(leave)	1.0
GH → gesture(stop)	1.0
ST → speech(take)	1.0
SL → speech(leave)	1.0
SH → speech(stop)	1.0
O → object	1.0
E → empty	1.0

Table 1: Example of a grammar with 3 intentions.

At the execution time, the fusion grammar $\mathbf{G_f}$ receives as input the results of the observations gathered with the recognition modules in the form of a triple $(\mathbf{g}, \mathbf{s}, \mathbf{o})$ representing, respectively, gestures, speech, and objects. This way, the Fusion Engine exploits a grammar parser in order to extract an intention and the associated probability. If one or more derivations are possible, it returns a N-best list of possible intentions in order of probability.

Rules	Score
S → T O	0.25
S → T E	0.25
S → L E	0.25
S → H E	0.25
T → GT ST	0.2
T → E ST	0.3
T → GP ST	0.4
T → GL E	0.1
L →GL SL	0.5
L → GL E	0.2
L → E GS	0.3
H → GH SH	0.2
H →GH E	0.5
H → E SH	0.3
GT → gesture(take)	1.0

GL → gesture(leave)	1.0
GH → gesture(stop)	1.0
GP → gesture(point)	1.0
ST → speech(take)	1.0
SL → speech(leave)	1.0
SH → speech(stop)	1.0
O → object	1.0
E → empty	1.0

Table 2: example of a trained grammar with 3 intentions

In Table 2, we illustrate an example of a grammar where the intentions are described as follows:

- The intention **take** is represented by the symbol T,
- The intention **leave** is represented by the symbol L
- The intention **stop** is represented by the symbol H.

Assume that the Fusion Engine receives the following triples:

1. *gesture(point) speech(take) an object*
2. *gesture(take) speech(stop) empty*
3. *gesture(leave) empty empty*

The probability of the first triple is 0.4 x 0.25, i.e. the product of the rules belonging to derivation tree (see Figure 7). As for the second triple, the parser can recognize that no derivations are possible in the grammar for that triple. In the last case, we have two possible interpretations, one for **take** (Figure 8a) and one for **leave** (Figure 8b), therefore the Fusion Engine returns the N-best list of intentions in order of probability. The parsing algorithm is Cocke-Younger-Kasami (CYK) [Kasami65].

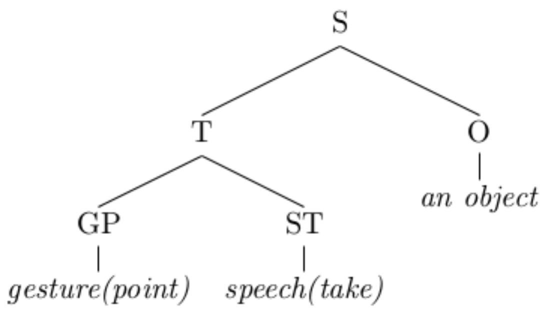


Figure 7: Derivation tree for gesture(point) speech(take) and an object on the scene

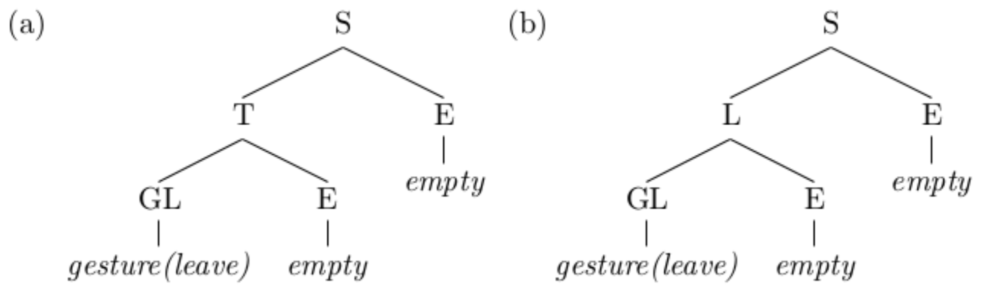


Figure 8: Derivation tree for an ambiguous intention a) gesture leave recognized as take intention b) gesture leave recognized as leave intention.

The FrameBuilder generates the output given a triple received in input. If the triple misses some arguments, the FrameBuilder waits for a fixed amount of time, if the new triple is compatible with the old one, this is maintained by default, otherwise, the new triple is considered as a separated data.

Gestures, Speech, and Fusion

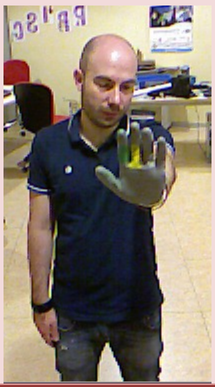
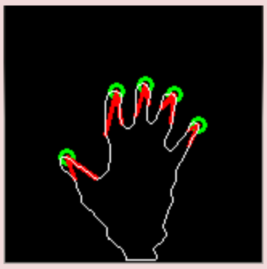
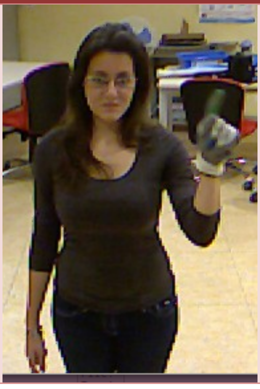
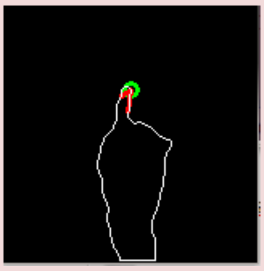
In this section, we detail the concrete repertory of gestures and speech commands we exploited for human-robot multimodal interaction in the benchmarking domains.

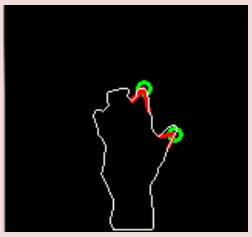
Basic set of gestures

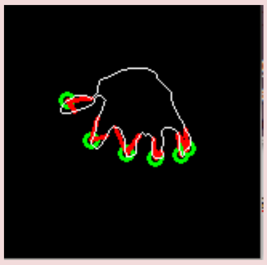
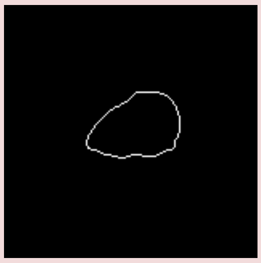
We start introducing an initial and general set of gestures suitable for interacting with a mobile robot that can handover objects with a human co-worker:

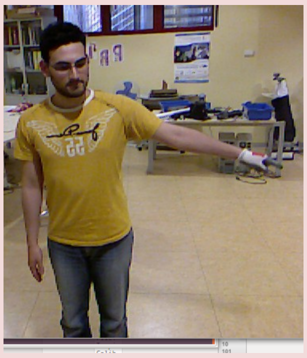
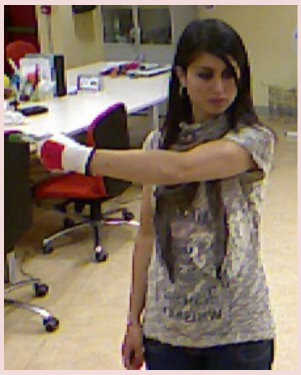
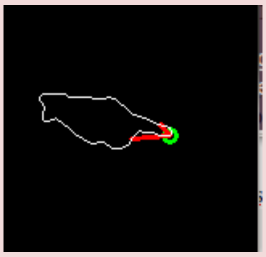
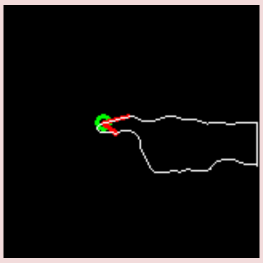
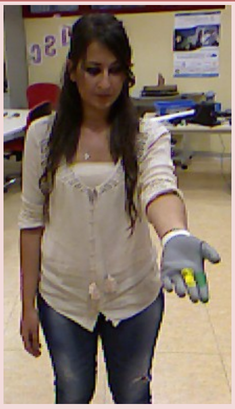
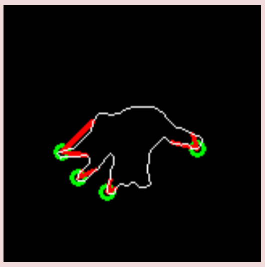
- Stop
- No
- Come
- Give
- Take\Leaves
- Point

In the following we describe each of these gestures.

Stop Gesture		
Users perform stop gesture raising their hand, usually just above the shoulder, and holding it thrust forward. In a base domain this gesture was intended to stop an entire run, a plan or to try to stop the robot immediately.		
No gesture		
The No gesture is performed with hand in pointing position and the hand or the pointing finger moving from right to left or vice versa. In the base domain it is designed to deny the last action received, if the wrong command is interpreted.		

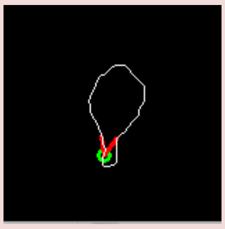
Come gesture		
The gesture « come » is performed with the hand of user raised more or less at the shoulder and fingers move by opening or closing the hand or moving the whole hand toward the user. Its function is to move nearer the robot to the position where the user is located.		

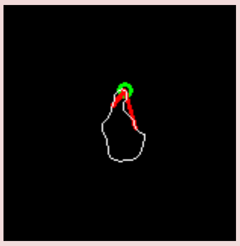
Take/Leave gesture		
User performs the « take » gesture by holding the hand open in front of him and then closing it. The gesture « leaves » runs in the same way but with the closed hand and then open hand. In a context of continuous gestures recognition it becomes difficult to classify these 2 gestures separately. Hence, we interpret it as a single gesture to be discriminated by the context. If the robot already holds the object the user wants, the gesture will be interpreted as « leave ». Otherwise, the gesture will be interpreted as « take ».		 

Point at gesture		
The « point at » gesture may indicate a location or a particular object. It is performed by pointing at a location or an object.	 	 
Give gesture		
The « give » gesture is performed by keeping the open hand stretched forward in order to receive an object from the robot.		

Airbus Use Case Gestures Domain

In the Airbus scenario the robot helps human users in assembling components of the Ariane shuttle. The initial set of gestures has been adapted to this scenario by adding two gestures, respectively for signaling the success of the operation performed by the robot and to abort the last executed operation. The latter gestures have been labeled respectively as <<ok>> and <<abort>>, and they correspond to a gesture with a thumb up and thumb down by the users.





DLR Use Case Gestures Domain

In order to address the DLR Use Case two new gestures, respectively for changing the modalities from the "teach mode" to the "execution mode" and from the "execution mode" to the "teach mode" have been added. The gesture of the <<switch>> is performed by moving the open hand from right to left or vice-versa.

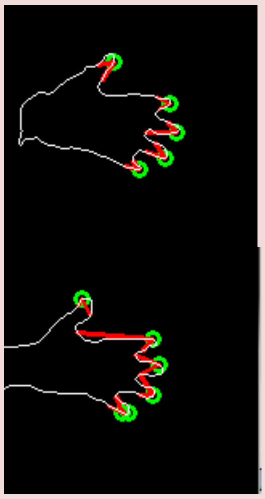


Table 3 illustrates the confusion matrix for all gestures. Test results have been obtained with a ten fold cross validation. Dataset consists of 16 different users and 140/160 occurrences of each gesture. The precision of the LDCRF classification is 0.72.

	Give	Stop	Point	pointDX	pointDx	No	Take	Come	Abort	Ok	Switch	Precision	Recall	F1
Give	108	14	7	6	0	0	7	0	4	0	1	0.71	0.73	0.72
Stop	26	92	5	0	0	1	6	4	0	19	2	0.53	0.59	0.56
Point	2	18	84	0	0	1	33	0	1	1	0	0.68	0.60	0.64
PointDX	0	0	1	137	14	0	0	0	0	0	0	0.80	0.90	0.85
PointSX	0	0	2	1	109	17	0	0	1	0	11	0.69	0.77	0.73
No	0	1	0	0	0	153	0	21	8	0	0	0.73	0.84	0.78
Take	9	20	20	5	0	0	109	8	0	6	8	0.66	0.59	0.62
Come	6	13	0	0	7	16	0	142	8	1	2	0.76	0.73	0.74
Abort	1	0	0	0	6	21	1	8	167	1	1	0.85	0.81	0.83
Ok	0	6	2	17	3	1	4	1	5	150	32	0.82	0.68	0.74
Switch	1	11	3	6	18	1	4	3	2	5	158	0.73	0.75	0.74
												0.72	0.73	0.72

Table 3: Confusion Matrix

Speech-based Interaction

The speech modality permits a rich interaction with the robot that covers all the commands previously introduced. The possible communication sentences are specified by a simple grammar, where each command is composed of three elements:

$$Command \leftarrow Keyword\ Action\ Information$$

In our domain, *Keyword* represents the actor, i.e. in our scenario the *robot*. *Action*s are in the following set: *take, leave, come, give, no, stop, ok, abort, change mode, open grip, close grip*. *Information* represents the target and can be instantiated for example by the following values: *glue, bottle, plate* and *screwdriver, hammer, yardstick, pincers*.

Multimodal Interaction

As already presented above, a Probabilistic Context Free grammar (PCFG) is used for multimodal fusion. In our scenarios, we assume that the intention is recognized given a triple (g, s, o):

$$Intention \leftarrow g\ s\ o$$

Where g belong to the set of labels of gestures, s belong to the set of actions recognized by the speech classifier, and the information is a field that can contains the target (subject, object, or a location). The fusion grammar is generated from the training set as illustrated in the previous section. Specifically, in the DLR and AIRBUS domains the intentions associated with the possible multimodal instances are the following:

- Take:
    - Gesture(take) speech(take) object_on_table
    - Gesture(take) object_on_table
    - Gesture(point) speech(take)
    - Speech(take) object_on_table
- Leave:
    - Gesture(take) speech(leaves) object_in_robot_hand
    - Gesture(take) object_in_robot_hand
    - Speech(leaves) object_in_robot_hand
    - Speech(leaves)
- Give:
    - Gesture(give) speech(give) object_in_robot_hand
    - Gesture(give) object_in_robot_hand
    - Speech(give) object_in_robot_hand
- Stop:
    - Gesture(stop) speech(stop)
    - Gesture(stop)
    - Speech(stop)
- No:
    - Gesture(no) speech(no)
    - Gesture(no)
    - Speech(no)
- Change Mode:
    - Gesture(switch) speech(switch)
    - Gesture(switch)
    - Speech(switch)
- Confirm:
    - Gesture(ok) speech(ok)
    - Gesture(ok)
    - Speech(ok)
- Abort:
    - Gesture(abort) speech(abort)
    - Gesture(abort)
    - Speech(abort)
- Come Closer:
    - Gesture(come) speech(come) location
    - Gesture(point) speech(come)
    - Gesture(come) location
    - Speech(come) location

Experimental Scenarios

The overall system has been implemented in the AIRBUS and the DLR scenarios. In the following we describe the domains along with possible enabled interactions.

AIRBUS Scenario and UNINA-LAAS Integration

In the AIRBUS domain [Ala14] the robot is to help the human co-worker for the bracket installation on the Ariane 5 front skirt. The bracket gluing procedure is composed of the following phases: (a) get the work-kit, (b) determine the right position of bracket, (c) prepare surface with adapted sand paper, (d) clean surface with an adapted cleaner, (e) wait evaporation, (f) apply adapted primer, (g) wait drying, (h) apply glue, (i) insert bracket. The robot must be able to recognize whether the user is asking for something or is directly performing an operation. A scenario similar to the AIRBUS domain has been implemented at LAAS-CNRS in order to test the human-robot interaction system. In this setting, a human operator is to interact with a PR2 robot for bracket installation. The human can exploit the multimodal interaction system described above. Here, objects and target locations of the bracket are detected using a bar code (see Figure 9).

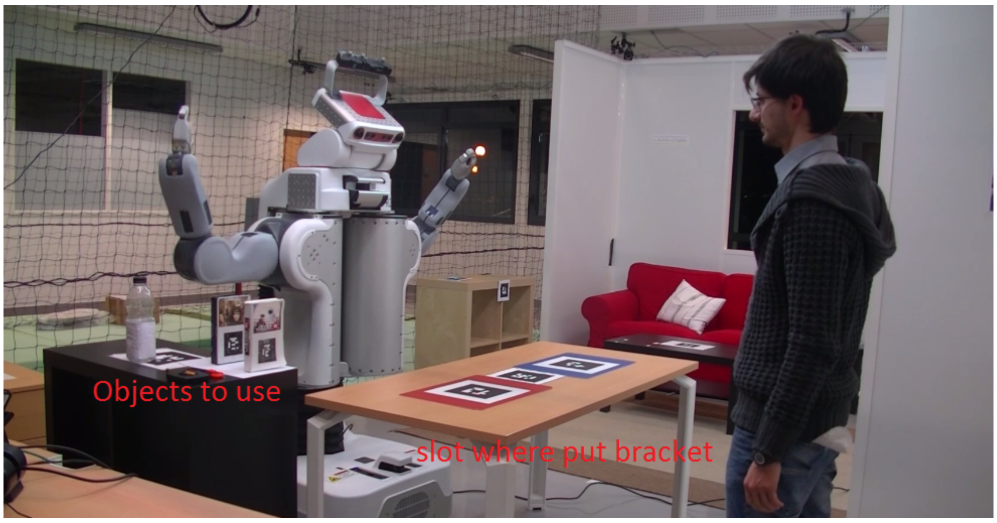


Figure 9: AIRBUS set-up at LAAS-CNRS

During the task execution the robot is to wait for multimodal human requests and to guide the human actions by pointing the position of the bracket.

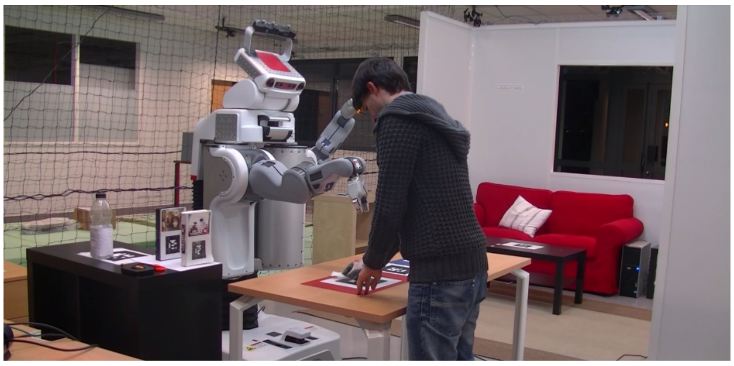


Figure 10: The human cleans the slot

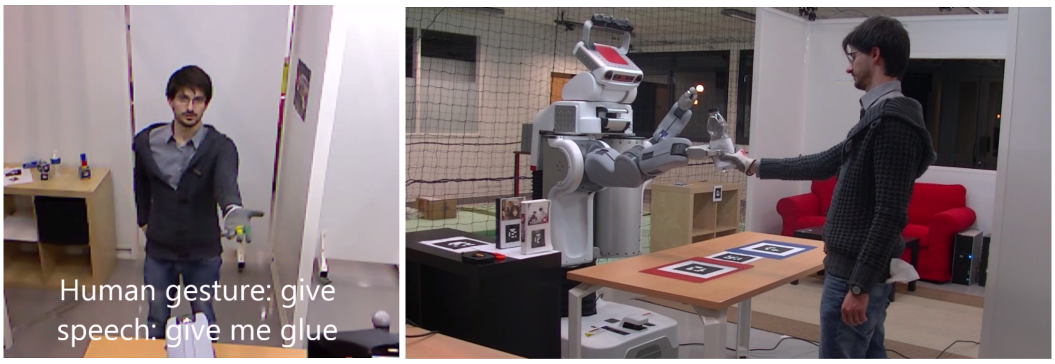


Figure 11: The human receives the object from the robot

After the cleaning phase, the user asks the robot for the glue using both gesture and speech. The intention of the user is recognized as a request for the glue by selecting the following interpretation:

$$give(glue) \ \leftarrow \ gesture(give)\ speech(give)\ object(glue)$$

The resulting robot action, see Figure 11, is to fetch the glue bottle from the work-kit to the human user and to start again pointing at the slot.

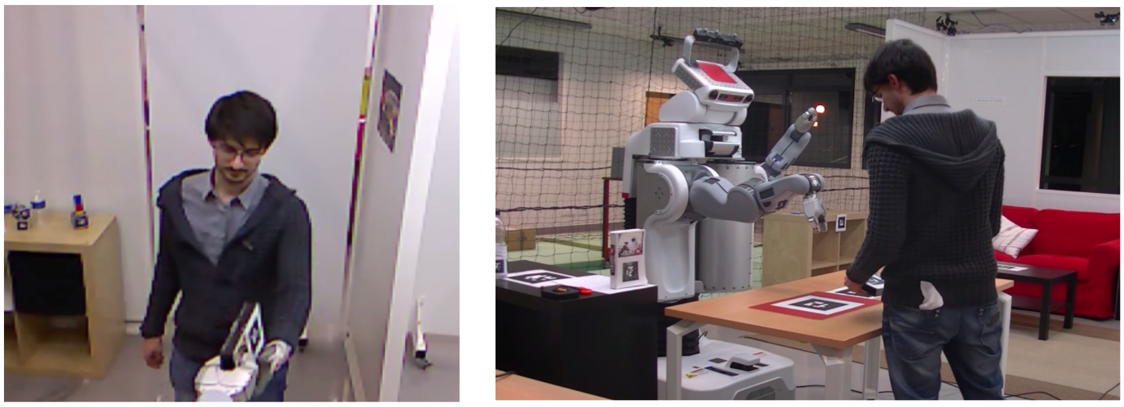


Figure 12: The human receives the bracket (left) and the robot guides bracket installation pointing towards the slot

When the glue is applied, the user asks the robot to receive the glue bottle from his hand. When the action is performed the robot proactively takes the bracket and gives it to the human (Figure 12).

DLR Hospital Scenario and UNINA-TUM Integration

The DLR Hospital domain concerns the process of quality control, preparation and packaging in a hospital center. In this context, the Human and the robot are both involved in the task of sorting instruments on a tray, interacting as specified in D8.4.1. This task is composed of five steps (see Figure 13): (A) the human user brings a tray of unsorted instruments to the robot; (B) he/she checks the instruments and puts them in a specified zone which is reachable by the robot; (C) the human asks the robot to help him in the task; (D) during the interaction the robot manipulates specific objects while the user checks the instruments and orders the other objects; (E) the task is completed when the tray contains ordered and checked instruments.

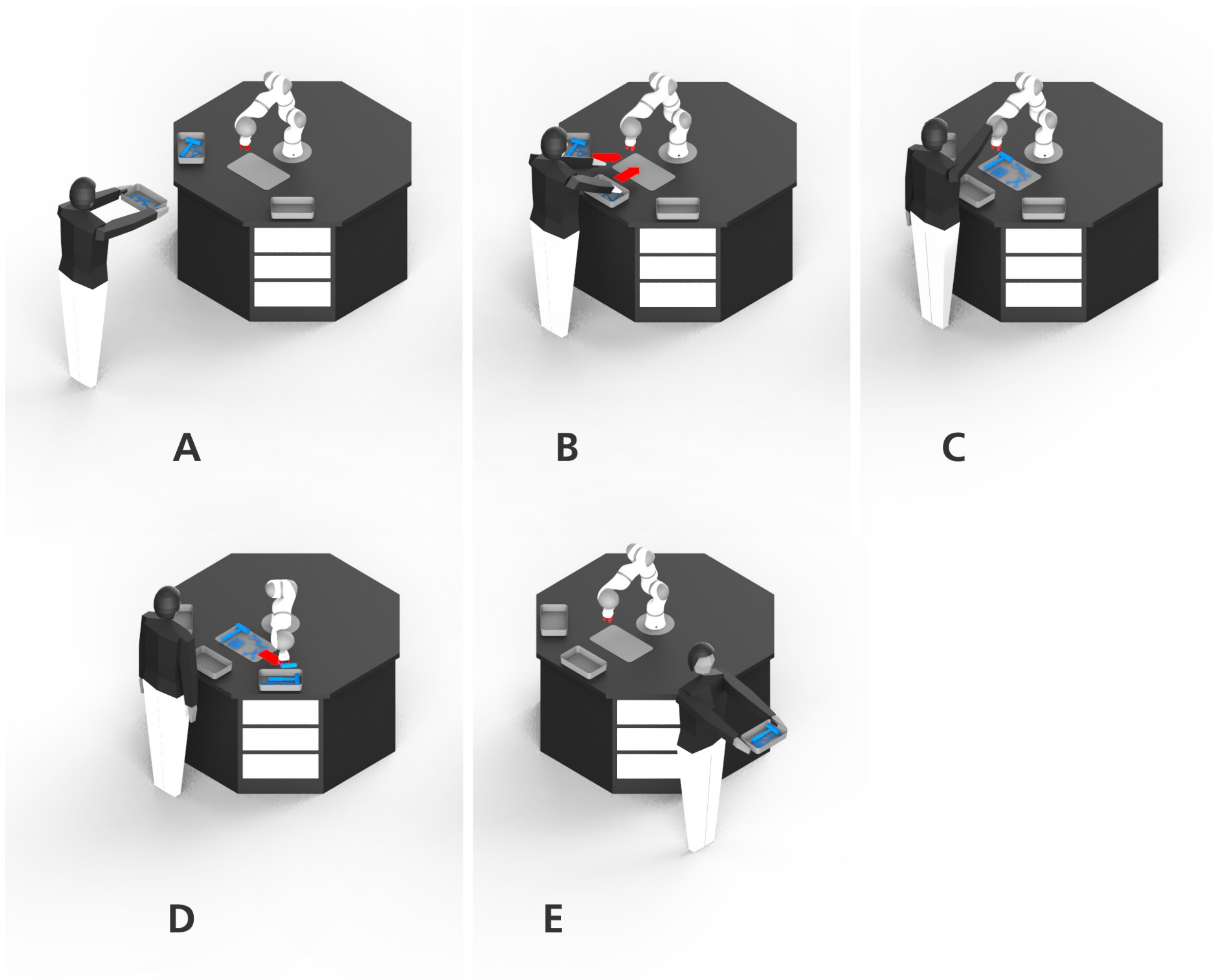


Figure 13: DLR Hospital domain and main phases of the interaction

In order to test the multimodal framework in this domain, we defined a similar set-up at TUM where the user and the robot interact in order to place specific tools in certain boxes. In this case, the robot already knows the target location of some tools, while the human should instruct the robot where (i.e., in which box) to place the other objects thought kinesthetic teaching. As already mentioned above, a specific gesture is introduced to switch from the teaching to the execution mode and vice-versa.

As a set-up, we considered a KUKA arm endowed with a gripper operating in a workspace monitored by two *kinect* cameras, one for user interaction and another used to track the objects on the table, which are recognized via the *qr code*. The setting includes two trays and three objects: a screwdriver, a yardstick (the unknown object) and a tape (see Figure 14).

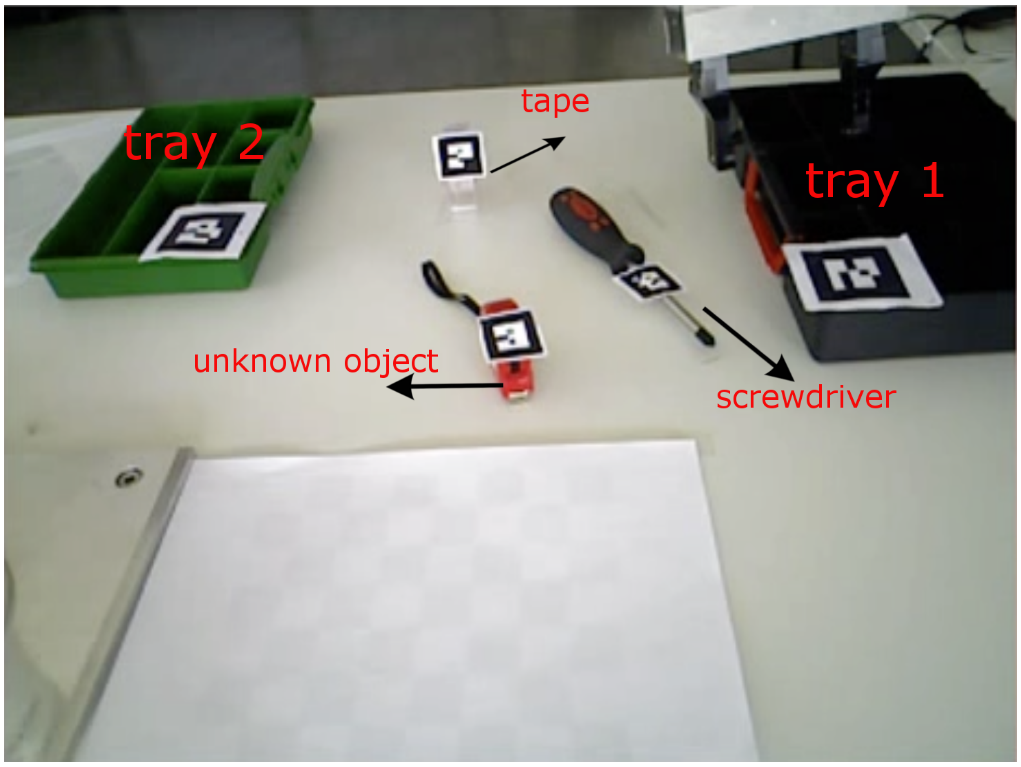


Figure 14: Tools in the TUM-UNINA integrated scenario

During the interaction, the human user is always enabled to switch to the teaching mode. The switch can be invoked in a multimodal manner using either a gesture or a voice command. After the switch, the robot goes in gravity compensation waiting for the human physical guidance. The operator first shows the target object to the robot (kinect), then, he/she exploits kinesthetic teaching moving the robot from the object position to the target. In this phase, commands of open/close gripper can be provided through the speech mode. Indeed, since the human physically interacts with the robot, gesture-mode can be uncomfortable or not possible (see Figures 15 and 16).

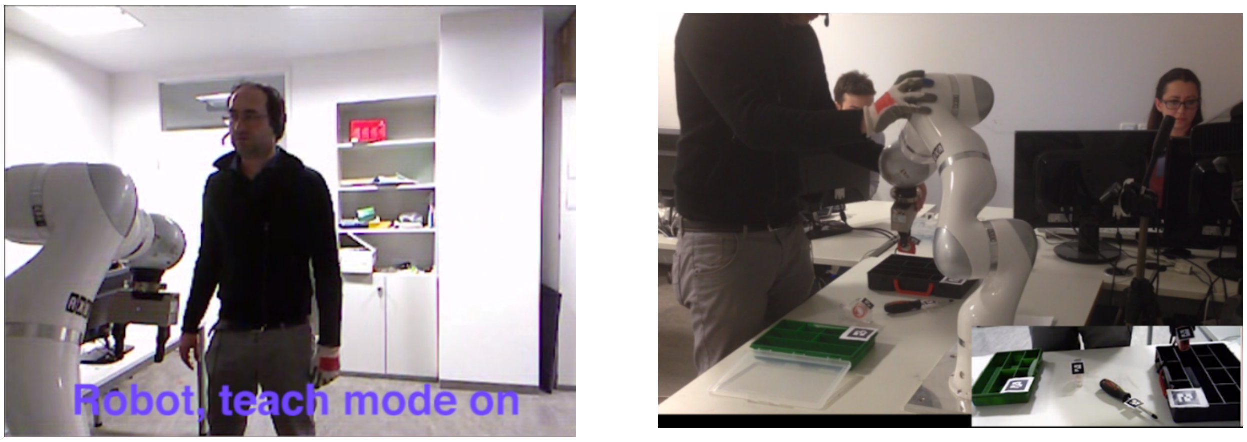


Figure 15 Teaching mode.

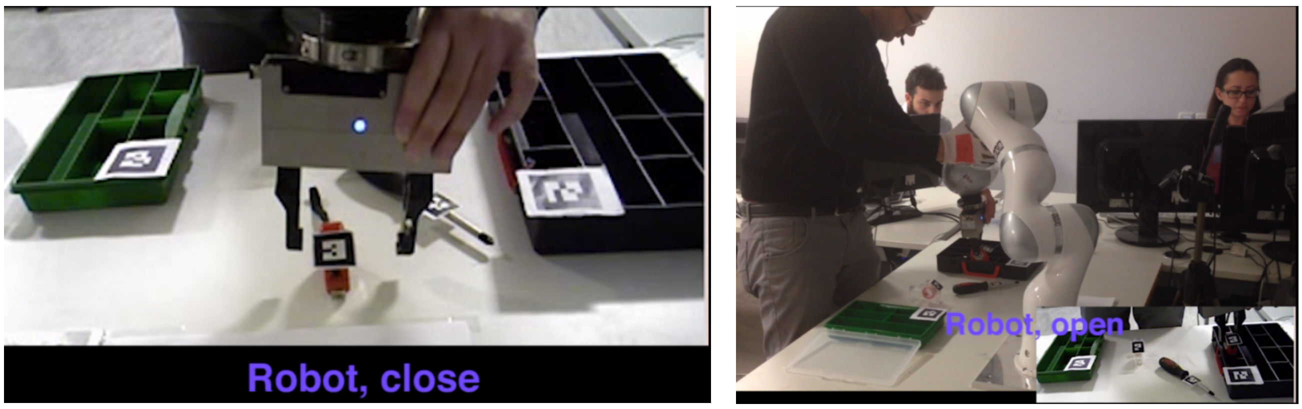


Figure 16: Teaching Mode: Speech-based interaction during the physical interaction

Before stopping the teaching mode, the user can ask the robot to repeat the learned sequence to verify its execution. In this context, the "take" gesture is used for this purpose (see Figure 17).

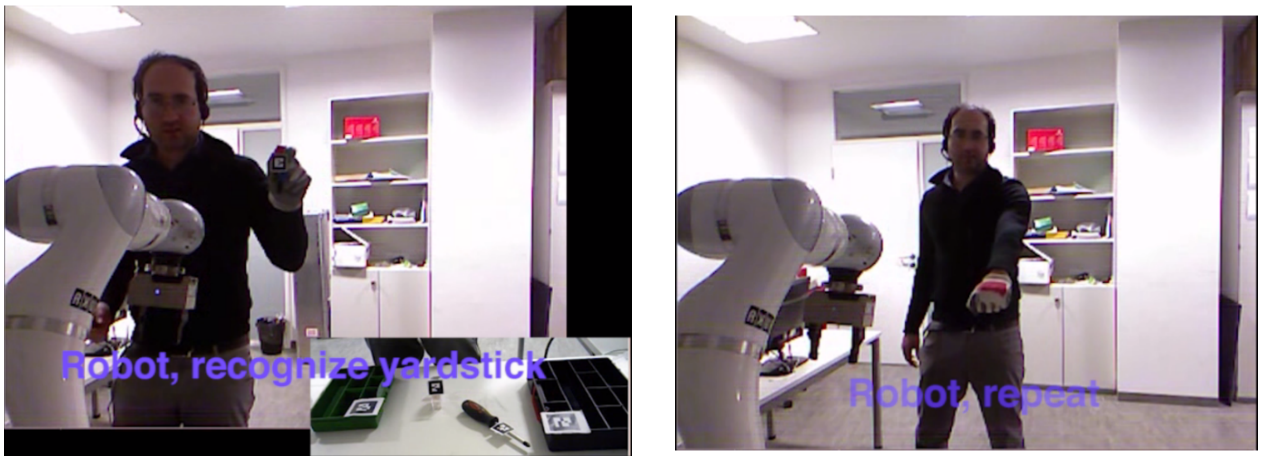


Figure 17: The human shows the target object (left) and asks for task repetition (right)

Summary

In this report, we presented the overall multimodal interaction framework focusing on gesture classification, multimodal fusion and interpretation of the human intentions in the context of the interaction. In particular, we described the deployment of LDCRF for continuous gesture recognition and Probabilistic Context Free Grammars (PCFGs) for multimodal fusion and interpretation. We described the implementation of the proposed framework in two concrete case studies: the AIRBUS domain and DLR Hospital domain. In these contexts, we described the associated repertories of recognized gestures and patterns of multimodal interaction. Finally, we illustrated experimental scenarios where the proposed framework is deployed for cooperative human-robot task execution.

References

[Allen83] J. F. Allen: "*Maintaining knowledge about temporal intervals*". In: *Communications of the ACM, ACM Press,* 1983.

[Cutugno13] F. Cutugno, A. Finzi, M. Fiore, E. Leone, S. Rossi, "Interacting with robots via speech and gestures, an integrated architecture", in Proceedings of INTERSPEECH 2013 - 14th Annual Conference of the International Speech Communication Association pp. 3727-3731. ISSN 2308-457X.

[Kasami65] T. Kasami, "An efficient recognition and syntax-analysis algorithm for context-free languages" (Technical report). Air Force Cambridge Research Laboratories, pages 65-758, 1965.

[Lafferty01] J. D. Lafferty, A. McCallum and F. C. N. Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". In Proceedings of the International Conference on Machine Learning, pages 282-289, 2001.

[Lee09] A. Lee and T. Kawahara. "Recent Development of Open-Source Speech Recognition Engine Julius". Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2009.

[Manning99] C. Manning and H. Schütze, "Foundations of Statistical Natural Language Processing", MIT Press. Cambridge, MA: May 1999.

[Morency07] L. P. Morency, A. Quattoni and T. Darrell. "Latent-dynamic discriminative models for continuous gesture recognition". In Proceedings of Computer Vision and Pattern Recognition, pages 1-8, 2007.

[Rosenfeld73] A. Rosenfeld, E. Johnston Angle detection on digital curves. IEEE Transactions on Computers, pages 875-878, 1973

[Rossi13] S. Rossi, E. Leone, M. Fiore, A. Finzi and F. Cutugno, "An Extensible Architecture for Robust Multimodal Human-Robot Communication", in Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) November 3-7, 2013. Tokyo, Japan, pp. 2208-2213. ISBN: 978-1-4673-6357-0/13.