

ROBotic Open-architecture Technology for
Cognition, Understanding and Behavior



Project No. 004370

RobotCub

Development of a Cognitive Humanoid Cub

Instrument: Integrated Project
Thematic Priority: IST - Cognitive Systems

D6.3 Interaction Histories: Software Module Notes WP6 - Gesture Communication

Due date: **01/03/2008**
Submission Date: **15/04/2008**

Start date of project: **01/09/2004**

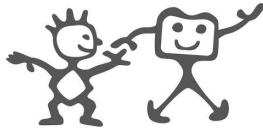
Duration: **60 months**

Organisation name of lead contractor for this deliverable: **University of Hertfordshire**

Responsible Person: **Kerstin Dautenhahn**

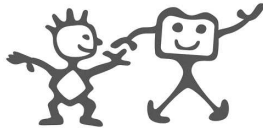
Revision: **1.0**

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	PU
PP	Restricted to other programme participants (including the Commission Service)	
RE	Restricted to a group specified by the consortium (including the Commission Service)	
CO	Confidential, only for members of the consortium (including the Commission Service)	



Contents

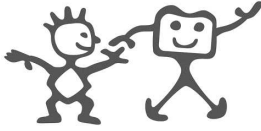
1	Executive Summary	3
2	IHA Process Diagram	4
3	Software Module Notes	5
3.1	Modules	5
3.1.1	Data Store	5
3.1.2	Kaspar2 Control	6
3.1.3	Kaspar Action Selection Process	7
3.1.4	Send Action Utility	8
3.1.5	Motivation Dynamics	8
3.1.6	Sound Sensor	8
3.1.7	Face Detector - IHA modifications to opencv_facedetect	9
3.1.8	iCub Control	9
3.1.9	iCub Action Selection Process	10
3.1.10	Sensors File Writer	11
3.1.11	Sensors File Reader	11



1 Executive Summary

WP6 continues to focus on interaction dynamics of social interaction during robot-human play and the prerequisites for gesture and non-verbal communication between robots and humans, as well as the realization of these capabilities in a robot. In this particular research work, techniques for achieving this capability in an autonomous robot through grounded sensorimotor experience and interaction histories, are investigated.

This deliverable supports a submission of code into the iCub repository that consists of an implementation of the “Interaction History Architecture” (see D6-3 for a description) in a YARP framework along with control code for both the Kaspar2 robot of the University of Hertfordshire as well as the iCub (in simulation).



2 IHA Process Diagram

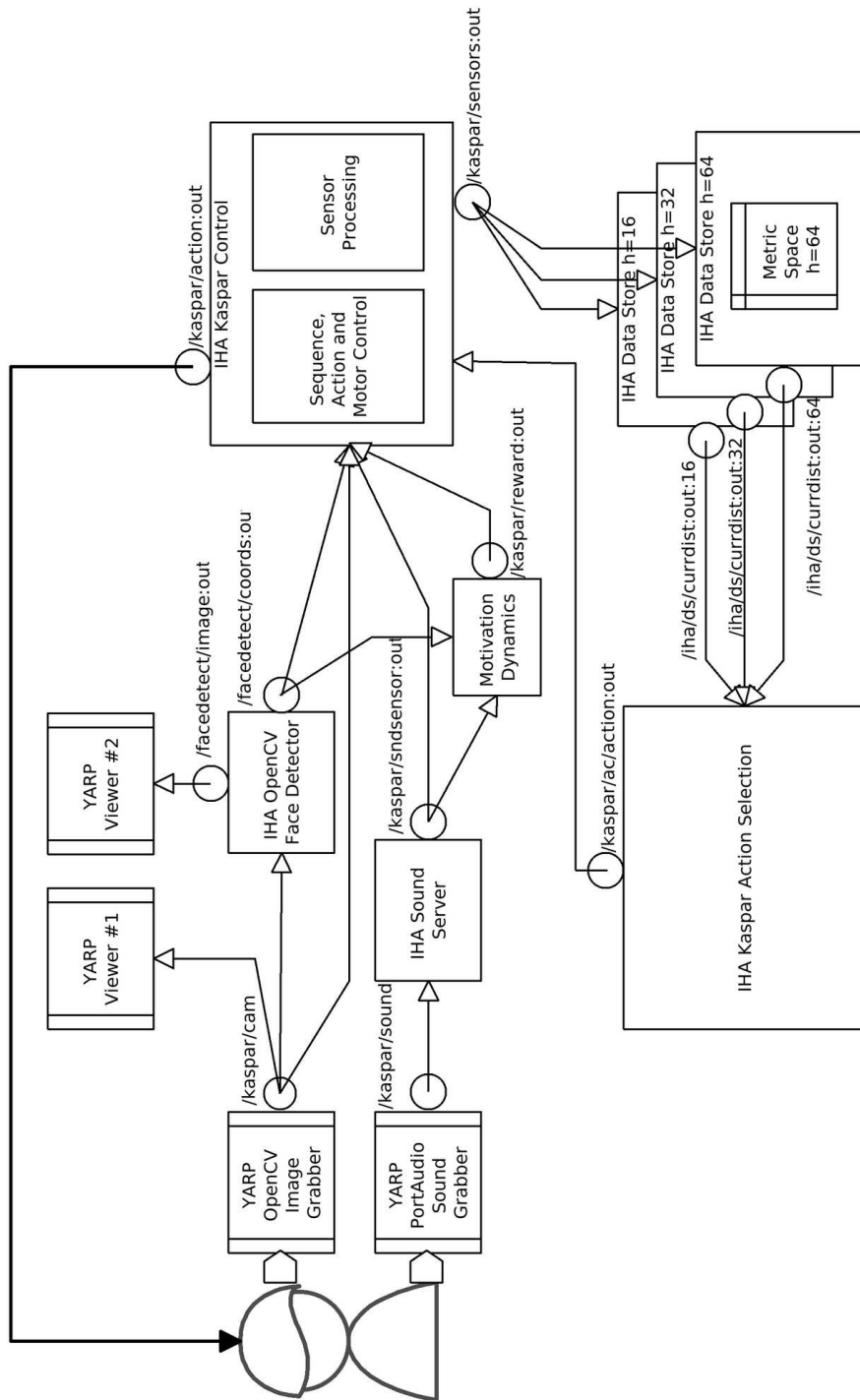
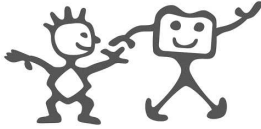


Figure 1: IHA Process Diagram showing main processes and connections.



3 Software Module Notes

3.1 Modules

3.1.1 Data Store

Description: The Data Store collects sensor data and creates experiences, placing them in a metric space and associating quality values and action values to create the interaction history space.

Executable: data_store.cpp

Files: data_store.cpp Main source file

experience/DataFrame.h Class to store a single data frame

experience/Experience.h Class to store a single experience

experience/ExperienceProcessor.h Processing functionality for experiences *e.g.* merge/delete them.

experience/ExperienceProcessor.cpp

experience/DistanceSpaceClass.h Class to hold the Distance Space and processing functionality at the Distance Space level.

experience/DistanceSpaceClass.cpp

experience/BinWindowMaxEntropy.h Adaptive binning using entropy maximization

experience/WindowIDCalc.h Information Distance calculation (Moving Window)

experience/ExperienceProcessorFileRW.cpp Read-Write Experiences

experience/serialization.h Serialization code

Execution Parameters: `-file <file>` configuration file

`-dbg <N>` Debug level (0-60)

`-save <file>` Save Experience Space when finished

`-load <file>` Load Experience Space from `!file!` before start

`-connect_to_sensors <port>` Connect to the specified sensor port on startup

Configuration Options: To be set in configuration file

name *String* Base name for ports (default iha)

dsnumber *Int* Data Store Number for multiple data stores (default 1)

HORIZONS *Int Int ...* List of horizons to keep metric spaces for

num_bins *Int* Number of bins in quantization. (default 5)

granularity *Int* Rate at which experiences are created in timesteps (default 1)

experience_action_gap *Int* For tuning correct association of action with experience (default 1, *i.e.* next action)

regular_experiences *String* Experiences created depending on timestep (default TRUE)

action_experiences *String* Experiences created when action changes (default FALSE)

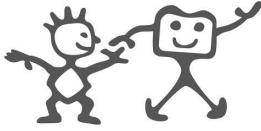
value_experiences *String* Experiences created when reward value changes (default FALSE)

num_actions *Int* Number of actions configured (default 4)

write_curr_dist_to_port *String* Current Distance list written to a port (default TRUE)

write_max_dsp_neighbours *Int* Number of neighbours to output (default 0)

write_max_dsp_radius *Double* Max radius of experiences in neighbour list (default 0)



neighbour_radius *Double* Neighbourhood size (default 1.0)
merge_adapt_type *String* Merge Adaptation Type NONE, CYCLE_TIME, NUM_COMPARISONS (default NONE)
merge_threshold *Double* Adaptive Merge Threshold (default 0.0)
merge_increment *Double* Adaptive Merge Increment (default 0.01)
only_merge_same_actions *String* (default FALSE)
merge_exp_threshold *Int* For NUM_COMPARISONS Merge Adaptation (default 400)
merge_cycle_time_threshold *Int* For CYCLE_TIME Merge Adaptation (default 400)
purge_experiences *String* Purge Experiences switch (default FALSE)
purge_threshold *Double* To purge only experience with quality less than this (default 0.0)
adaptive_binning *String* Adaptive Binning using Entropy Maximization (default FALSE)
adaptive_binning_window_size *Int* Adaptive Binning - window over which entropy is maximized (default 32)
histogram_resolution *Int* Adaptive Binning (default 256)
future_horizon *Int* Horizon over which quality is updated (default 200)
future_value_update_type *String* Can be MAX, or BIASED (default MAX)
metric_space_heuristic *String* Can be NONE, TREE or NEIGHBOUR (default NONE)
verify_heuristic *String* For testing (default FALSE)
heuristic_start_threshold *Int* For Neighbour Heuristic algorithm (default 40)
heuristic_tree_radius *Double* For Tree heuristic algorithm (default 1.0)
num_image_sensors_x *Int* Number of image sensors to make from image - X direction (default 8)
num_image_sensors_y *Int* Number of image sensors to make from image - Y direction (default 8)
use_reward_action_in_exp *String* Whether experience includes the reward and action as sensors (default TRUE)

Ports Created: /<name>/ds<dnumber>/data:in Input port

/<name>/ds/currdist:out:<horizon> Output port for current experience neighbours

3.1.2 Kaspar2 Control

Description: Control for the Kaspar2 Robot and Sensor Collector. As well as providing the `sendAction()` function for the Kaspar2 robot, this module reads all necessary sensor data including image data, motivation feedback data, sound data and face detection data then consolidates them and writes them to a port.

Executable: kaspar/kaspar_control

Files: kaspar/kaspar_control.cpp Main source file

kaspar/KasparActions.cpp Class for holding kaspar action specifications

kaspar/KasparActions.h

kaspar/KasparSequence.h Class for holding kaspar motor control sequences for actions

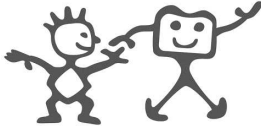
Execution Parameters: -file <file> configuration file

-hwconfig <file> hardware configuration file

-dbg <N> Debug level (0-60)

-connect_to_image <port> Connect to the specified image port on startup

-connect_to_coords <port> Connect to the specified port for detected face coordinates on startup



- connect_to_reward** <port> Connect to the specified port for reward data on startup
- connect_to_soundsensor** <port> Connect to the specified sound sensor port on startup

Configuration Options: To be set in configuration file

- name** *String* Base name for ports (default iha)
- action_defs_file** *String* File in which action definitions are configured (default action_defs.txt)
- num_image_sensors_x** *Int* Number of image sensors to make from image - X direction (default 8)
- num_image_sensors_y** *Int* Number of image sensors to make from image - Y direction (default 8)
- sensoratarate** *Int* Sensor data rate for output in ms (default 100)
- reward_display** *String* Display reward by using expressive actions (default TRUE)
- action_ehi** *Int* Action (expression) to execute for High reward (default 1)
- action_elo** *Int* Action (expression) to execute for Low reward (default 16)
- action_emid** *Int* Action (expression) to execute for Mid reward (default 2)
- th_ehi** *Int* High Threshold for expression change (default 0.8)
- th_elo** *Int* Low Threshold for expression change (default 0.3)

Ports Created: /<name>/ac/action:out Action Advice output port

- /<name>/sensor:out Sensor output port
- /<name>/action:cmd Action Reader input port
- /<name>/image:in Image input port
- /<name>/coords:in Face Coordinates input port
- /<name>/reward:in Reward input port
- /<name>/soundsensor:in Sound Sensor input port

3.1.3 Kaspar Action Selection Process

Description: Wrapper for the action selection process

Executable: kaspar_action_selection

Files: kaspar_action_selection.cpp Main source file

include/iCub/iha/action_selection_main_loop.h Generic action selection loop. This is the main process that takes in the nearest neighbour list and uses the roulette wheel action selection process to generate action advice.

kaspar/KasparActions.cpp Class for holding kaspar action specifications

kaspar/KasparActions.h

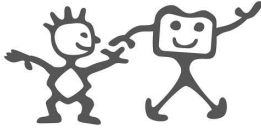
kaspar/KasparSequence.h Class for holding kaspar motor control sequences for actions

Execution Parameters: -file <file> configuration file

- dbg <N> Debug level (0-60)
- connect_to_action <port> Connect to the specified action port on startup
- connect_to_dist <port> Connect to the specified nearest neighbour distance port on startup

Configuration Options: To be set in configuration file

- name** *String* Base name for ports (default iha)
- action_defs_file** *String* File in which action definitions are configured (default action_defs.txt)
- neighbour_radius** *Double* Max radius of neighbourhood. (default 1.0)



temperature *Double* Starting temperature (affecting chance of random) (default 4.0)

temp_dec *Double* Decrement of temperature per action step (default 0.002)

Ports Created: /<name>/ac/action:out Action Advice output port

/<name>/ac/currdist:in:<horizon> Input port for current experience neighbours

3.1.4 Send Action Utility

Description: Utility to send an action to an active control process

Executable: send_action

Files: <control>/send_action.cpp Main source file

Execution Parameters: -file <file> configuration file

-dbg <N> Debug level (0-60)

-connect_to_action <port> Connect to the specified action port on startup

Configuration Options: To be set in configuration file

name *String* Base name for ports (default iha)

Ports Created: /<name>/ac/singleaction:out Action Advice output port

3.1.5 Motivation Dynamics

Description: Collects the sound sensor and face detection data and writes a resultant reward to a port

Executable: motivation_dynamics

Files: motivation_dynamics/motivation_dynamics.cpp Main source file

Execution Parameters: -file <file> configuration file

-dbg <N> Debug level (0-60)

-connect_to_coords <port> Connect to the specified port for detected face coordinates on startup

-connect_to_soundsensor <port> Connect to the specified sound sensor port on startup

Configuration Options: To be set in configuration file

name *String* Base name for ports (default iha)

Ports Created: /<name>/reward:out Reward output port

/<name>/coords:in Face Coordinates input port

/<name>/soundsensor:in Sound Sensor input port

3.1.6 Sound Sensor

Description: Creates a single valued sensor from a YARP sound stream

Executable: sound_sensor

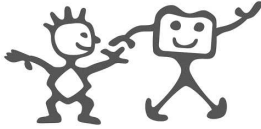
Files: sound/sound_sensor.cpp Main source file

Execution Parameters: -file <file> configuration file

-dbg <N> Debug level (0-60)

-connect_to_soundsensor <port> Connect to the specified sound sensor port on startup

Configuration Options: To be set in configuration file



name *String* Base name for ports (default iha)
soundSENSORrate *Int* Rate at which the sound sensor data is produced on the output port in ms. (default 100)
soundgain *Double* To compensate for low volume sound source. (default 2.5)

Ports Created: /<name>/soundSENSOR:out Sound Sensor output port
/<name>/sound:in Sound Stream input port

3.1.7 Face Detector - IHA modifications to opencv_facedetect

Description: Detects faces in YARP images on a port using multiple HAAR cascades. Chooses largest face if more than one is detected and outputs the coordinates on a YARP port.

Executable: facedetect

Files: iha.facedetect/face_detect.cpp Main source file

Execution Parameters: -file <file> configuration file
-dbg <N> Debug level (0-60)

Configuration Options: To be set in configuration file

PORTS *Group* Group level; List of ports. Requires definitions for input, output and coords ports.

CASCADES *Group* Group level; List of cascades.

Ports Created: Specified in config file. Opens an Input port for images, an Output port for images and an output port for Coordinates of detected faces.

3.1.8 iCub Control

Description: Control for the icub robot (ODE simulator currently) and Sensor Collector. As well as providing the sendAction() function for the iCub robot, this module reads all necessary sensor data including image data, motivation feedback data, sound data and face detection data then consolidates them and writes them to a port.

Executable: iCub/icub_control

Files: iCub/icub_control.cpp Main source file

iCub/ICubActions.cpp Class for holding iCub action specifications

icub/ICubActions.h

icub/ICubSequence.h Class for holding iCub motor control sequences for actions

Execution Parameters: -file <file> configuration file

-hwconfig <file> hardware configuration file

-dbg <N> Debug level (0-60)

-connect_to_image <port> Connect to the specified image port on startup

-connect_to_coords <port> Connect to the specified port for detected face coordinates on startup

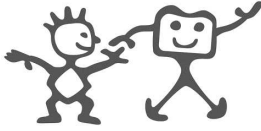
-connect_to_reward <port> Connect to the specified port for reward data on startup

-connect_to_soundSENSOR <port> Connect to the specified sound sensor port on startup

Configuration Options: To be set in configuration file

name *String* Base name for ports (default iha)

action_defs_file *String* File in which action definitions are configured (default action_defs.txt)



num_image_sensors_x *Int* Number of image sensors to make from image - X direction (default 8)

num_image_sensors_y *Int* Number of image sensors to make from image - Y direction (default 8)

sensordatarate *Int* Sensor data rate for output in ms (default 100)

Ports Created: /<name>/ac/action:out Action Advice output port

/<name>/sensor:out Sensor output port

/<name>/action:cmd Action Reader input port

/<name>/image:in Image input port

/<name>/coords:in Face Coordinates input port

/<name>/reward:in Reward input port

/<name>/soundsensor:in Sound Sensor input port

3.1.9 iCub Action Selection Process

Description: Wrapper for the action selection process

Executable: icub_action_selection

Files: **icub_action_selection.cpp** Main source file

include/iCub/iha/action_selection_main_loop.h Generic action selection loop. This is the main process that takes in the nearest neighbour list and uses the roulette wheel action selection process to generate action advice.

icub/ICubActions.cpp Class for holding iCub action specifications

icub/ICubActions.h

icub/ICubSequence.h Class for holding iCub motor control sequences for actions

Execution Parameters: **-file** <file> configuration file

-dbg <N> Debug level (0-60)

-connect_to_action <port> Connect to the specified action port on startup

-connect_to_dist <port> Connect to the specified nearest neighbour distance port on startup

Configuration Options: To be set in configuration file

name *String* Base name for ports (default iha)

action_defs_file *String* File in which action definitions are configured (default action_defs.txt)

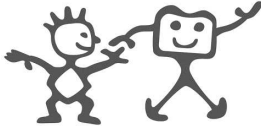
neighbour_radius *Double* Max radius of neighbourhood. (default 1.0)

temperature *Double* Starting temperature (affecting chance of random) (default 4.0)

temp_dec *Double* Decrement of temperature per action step (default 0.002)

Ports Created: /<name>/ac/action:out Action Advice output port

/<name>/ac/currdist:in:<horizon> Input port for current experience neighbours



3.1.10 Sensors File Writer

Description: Read Sensor data from a port and write to a file

Executable: `sensor_file_writer`

Files: `sensor_file_writer.cpp` Main source file

Execution Parameters: `-file <file>` configuration file

`-dbg <N>` Debug level (0-60)

`-connect_to_sensors <port>` Connect to the specified sensor port on startup

`-connect_to_image <port>` Connect to the specified image port on startup

Configuration Options: To be set in configuration file

`name String` Base name for ports (default iha)

`-write_data_to_file String` default TRUE

`-write_timestamp_images String` Images saved at every timestep (default TRUE)

`-write_all_images String` Images saved on reception (default FALSE)

`-out_dir_base String` Directory for output of images (default ../data)

`-sensor_file_name String` Name of sensor file (default sensors.out)

`-image_base_name String` Basename of image files (default image)

Ports Created: `/<name>/filewriter/img:in` Input port for images

`/<name>/filewriter/data:in` Input port for sensor data

3.1.11 Sensors File Reader

Description: Read Sensor data from a file and write sensor and image data to YARP ports replacing the robot

Executable: `sensor_file_reader`

Files: `sensor_file_reader.cpp` Main source file

Execution Parameters: `-file <file>` configuration file

`-dbg <N>` Debug level (0-60)

`-connect_to_ds <port>` Connect to the data store

`-dir <path_to_data_dir>` Data directory

Configuration Options: To be set in configuration file

`name String` Base name for ports (default iha)

`-read_sensor_data String` default TRUE

`-read_timestamp_images String` Images read from timestep files at every timestep (default TRUE)

`-read_all_images String` Images read from image files (default FALSE) - (*not working*)

`-sensor_file_name String` Name of sensor file (default sensors.out)

`-image_base_name String` Basename of image files (default image)

Ports Created: `/<name>/filewriter/img:out` Output port for images

`/<name>/filewriter/data:out` Output port for sensor data