RobotCub.org

ROBotic Open-architecture Technology for
Cognition, Understanding, and Behavior

Cogsys
Cognitive Systems

Project No. 004370

RobotCub

Development of a Cognitive Humanoid Cub

Instrument:	Integrated Project
Thematic Priority:	IST – Cognitive Systems

D7.3 Experimental results of tests with existing platforms

Due Date: Month 24
Submission date: Month 24

Start date of project: **01/09/2004** Duration: **60 months**

Organisation name of lead contractor for this deliverable: DIST, University of Genova

Responsible Person: Giorgio Metta

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)	

Table of Contents

1 Introduction

This deliverable item is a short summary that describes the existing platform we employed for experimentation within the RobotCub consortium. This information forms the basis of the compilation of the specifications of the iCub platform both in terms of mechanics, control and software. This deliverable also tries to make the presentation of the various stains of research (developed by various research groups on this diverse set of platforms) more coherent. It should also provide a clearer basis for understanding our work up to this point and until the iCub platform is ready to be shared for experiments. The deliverable is divided by partner.

2 Links

See also:

- Deliverable items: 7.1 and 8.3 both available online.

3 UGDIST

UGDIST has been working on two different humanoid robotic platforms. The following text is reported here for completeness although it might have appeared elsewhere in slightly different forms (published, as part of previous projects, etc.). The Babybot platform has been developed at LIRA-Lab, UGDIST since 1996, the software architecture in various formats is still available as Yarp version 1 (http://yarp0.sf.net) although a complete new re-engineered version has been produced for RobotCub. It is not recommended to use Yarp version 1. The new version is available from the same website and it is know as Yarp version 2. Details of the architecture are also reported in the deliverable item 8.3 (software architecture). Software is available open source from SourceForge, iCub specific software is available from the RobotCub website (see deliverable 8.3 for details).

This section describes the Babybot architecture in terms of hardware and supporting software. It includes definition of the hardware with particular emphasis to the robotic hand, which is crucial to the project. It contains also a description of the low-level software layer, a behavior based componentized software structure that provides robot control functionalities and communication within a distributed architecture based on Pentium class processors and running on a variety of operating systems. Further, this document includes the description of a learning model that to some extent has been repeatedly used within our architecture.

Overall, the robot architecture can be seen as a layered system as shown in Figure 1. It is worth stressing here layering involves only the “engineering” of the robotic artifact. It does not constrain what specific learning schema is employed or how the different developmental modules interact one with another. In this sense this is not in contrast with the main philosophical underpinning of the project: that is, certain cognitive skills have to be acquired through a prolonged developmental period. Although modularity (and layering in this case) is perhaps the only tool we have in face of complexity, we previously suggested that we need to carefully take into account how interrelationships between modules evolve as artificial adaptation unfolds if we are to properly characterize learning and development. It was important thus not to commit too early into the decision of which learning schema or control structure had to be used.

Bare robot hardware

Interface cards

OS independent software interface

Robot independent code

Robot specific interface

Learning architecture

Specific experiments

Figure 1 Organization of the hardware and software layers of the robotic artifact

The seven layers in Figure 1 span the conceptual space from the “bare metal” to the “experimental level”. All what we have designed in between is sort of “supportive” structure. These middle layers might condition, to a certain extent, whether a particular type of experimentation could be actually carried out, and consequently they represented a delicate design decision.

The first layer contains the robot hardware. It includes a simplified description of the mechanics and the limitations of the current implementation. The robot’s sensors and actuators are interfaced to a set of computers through interface cards. Layer number 2 is made of these cards. Layer 3 is the first software interface with the hardware. It is written with portability in mind since many different cards might be connected to the same setup [or, on the contrary, the same card might be used in another setup], the controlling computers could be possibly running different operating systems, and different subparts might be running on different machines connected through an IP-based network. Using a publicly available portable library (ACE), we designed a communication system and general purpose computation environment. Our guiding criterion was “try not to be monolithic”. In this sense our system does not force the use of any particular library or set of libraries and in fact it was easily interfaced to Intel IPL (for efficient image processing), and Matlab (because of the many available toolboxes). The core system runs seamlessly on Windows (NT series), QNX (a real-time OS), and Linux.

In general, code directly controlling the robot tends to have a nasty organization and often contain details of the specific locale even when the controlling cards are the same. We wanted to decouple the “robot” part of our software from the more general purpose OS interface. Consequently, layer 4 is meant as an interface between the robot-independent code built on a generic “device driver” definition [“device driver” here is a user-level software component]. The goal of the latter is of shielding the nasty details from the higher level where unspecific code could be produced. This allows interfacing to any different hardware at the price of rewriting a relatively small virtual device driver module. Layer 5 allows customization of the robot control code to the specific locale. This is required in order to “tune” the control code to the details of the actual robot. It allows reusing the same code on two not-too-dissimilar setups. It consists mainly on specific software and configuration files.

Layer 6 contains a general description of the organization of the learning modules of the system. These components are simply a tentative formalization of the implementation of the sensorimotor coordination behaviors of the robot. At this stage they are not necessarily innovative in any respect and in fact we pretty much relied on standard function approximation methods and traditional neural network algorithms (e.g. backpropagation).

Finally, layer 7 represents the level where specific experiments are conducted. It is worth noting that none of this organizational structure is strictly imposed. From the outside the architecture consists of a set of libraries and executable modules. The remainder of this document is divided into sections describing each layer to a reasonable level of detail.

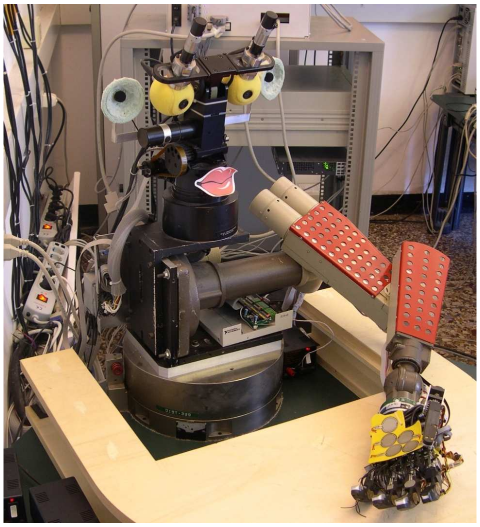


Figure 2: The robotic setup, the Babybot.

3.1 Babybot – the bare robot hardware

Babybot is an upper torso humanoid robot with a head, a manipulator arm and a hand. The head has five degrees of freedom (DoF). Three of them are associated with the two cameras to achieve independent panning and coupled tilting. The two remaining DoF allow panning and tilting at the level of the neck respectively. The manipulator is a 6 DoF Unimation PUMA 260, mounted with the shoulder horizontal to better resemble a human like arm (see Figure 2). In practice since it lacks a degree of freedom in the shoulder it often requires awkward configurations to reach a point in space.
Attached to the arm end point is a 5 fingered robot hand. Each finger has 3 phalanges; the thumb can also rotate toward the palm. Overall the number of degrees of freedom is hence 16. Since for reasons of size and space it is practically impossible to actuate the 16 joints independently, only six motors were mounted in the palm. Two motors control the rotation and the flexion of the thumb. The first and the second phalanx of the index finger can be controlled independently. Middle, ring and little finger are linked mechanically thus to form a single virtual finger controlled by the two remaining motors. No motor is connected to the fingertips; they are mechanically coupled to the preceding phalanges in order to bend in a natural way as explained in Figure 3.

The mechanical coupling between gears and links is realized with springs. This has the following advantages:

- The mechanical coupling between middle, ring, and small finger is not rigid. The action of the external environment (the object the hand is grasping) can result in different hand postures (see Figure 5).
- Low impedance, intrinsic elasticity. Same motor position results in different hand postures depending on the object being grasped.
- Force control: by measuring the spring displacement it is possible to gauge the force exerted by each joint.

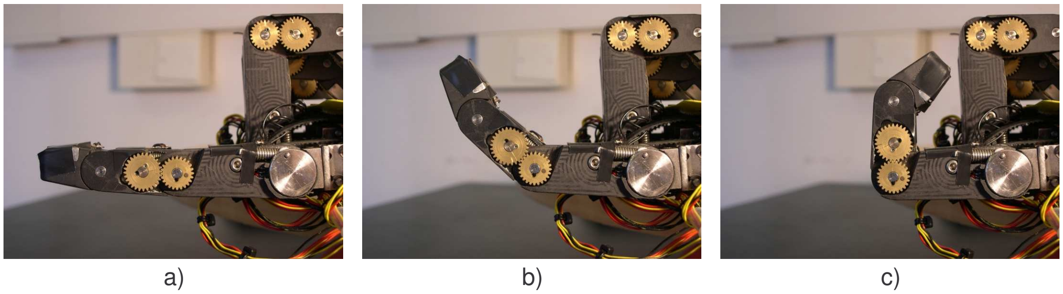


Figure 3: Mechanical coupling between the second and the third phalanges. The second phalanx of the index finger is directly actuated by a motor. Two gears transmit the motion to the third phalange. The movement is respectively of 90 and 45 degrees.

The robot’s sensory systems include vision, audition, touch, proprioception, and inertial sensing. Proprioceptive feedback is achieved by means of the motor optical/magnetic encoders. Two cameras rotating with the eyes and two microphones attached to the head respectively provide visual and auditory feedback. During the acquisition, images are sampled non-uniformly to mimic the distribution of receptors of the human retina. More pixels are acquired in the central part of the image (fovea) and less in the periphery (mathematically the distribution is approximated by a log-polar function).
The head mounts a three axis gyroscope that provides the robot with an artificial equivalent of the human vestibular system (in Figure 4). This sensor measures inertial information consisting of angular velocity along three orthogonal axes. It can be used for stabilizing the visual world efficiently and in coordinating the movement of the head with that of the eyes.

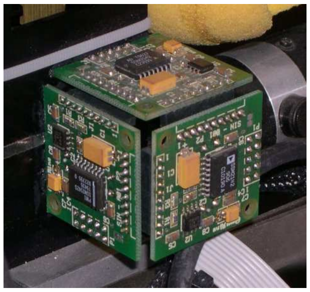


Figure 4: The inertial sensor of the Babybot developed at LIRA-Lab. It consists of three mono-axial sensors arranged along three orthogonal axes.

For the hand, Hall-effect encoders at each joint measure the strain of the hand's joint coupling spring. This information jointly with that provided by the motor optical encoders allows, at least in theory, estimating the posture of the hand and the tension at each joint. In addition, force sensing resistor (FSRs) sensors are mounted on the hand to give the robot tactile feedback. These commercially available sensors exhibit a change in conductance in response to a change of pressure. Although not suitable for precise measurements, their response can be used to detect contact and measure to some extent the force exerted to the object surface. Five sensors have been placed in the palm and three in each finger [apart from the little finger] (see Figure 4).

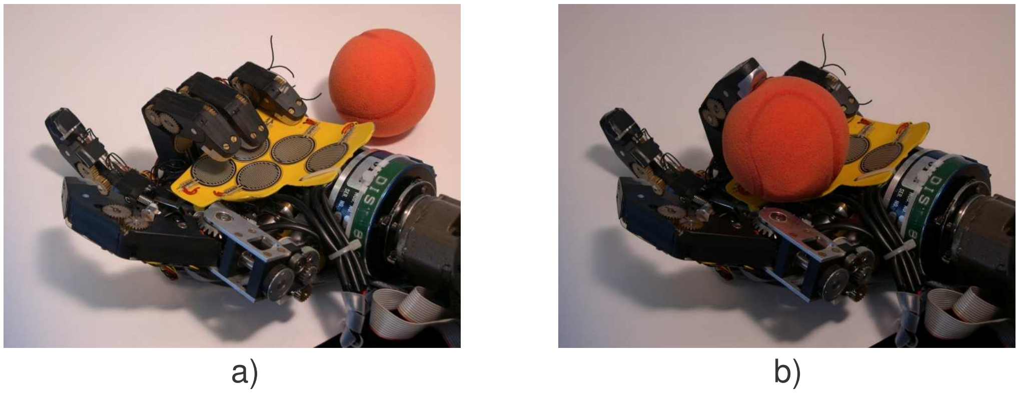


Figure 5: Elastic coupling. a) and b) show two different postures of the hand. Note however that in both cases the position of the motor shafts is the same. In b) the intrinsic compliance of the medium finger allow the hand to adapt to the shape of the object.

Further proprioceptive information is provided to the robot by a strain gauge torque/force sensor mounted at the link between the hand and the manipulator's wrist. This device is a standard JR3 sensor designed specifically for the PUMA flange. It can measure forces and torques along three orthogonal axes (see Figure 5).

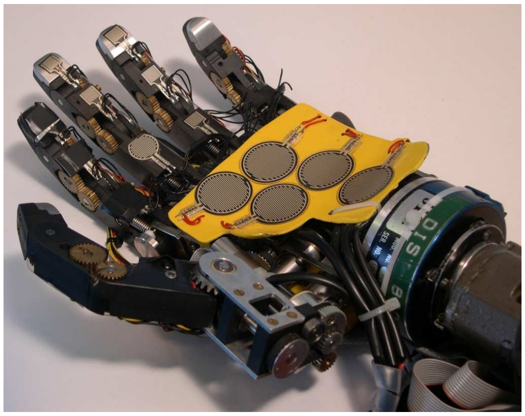


Figure 6: Tactile sensors. 17 Sensors have been placed: five in the palm, three on each finger apart the little finger. In this picture the sensors in the thumb are hidden. The short blue cylinder that links the PUMA wrist to the hand is the J3R force sensor.

3.2 Interface cards

The robot sensing includes some digitizing interfaces and special signal conversion and conditioning modules. The link between the hardware and the robot is provided through standard PCI/ISA cards, serial ports, etc.

Motor control also requires special hardware to generate the appropriate signal driving the motors. At this level the Babybot follows a very traditional electric motor control approach. The robot is actuated by DC motors. All of them have their specific control card and power amplifier. In the case of the PUMA [the arm] the original (the one provided by Unimation) linear amplifier was modified and interfaced to the standard control card on board a PC. The head and hand joints are controlled through a bank of switching amplifiers (PWM). Each control card has a DSP on board and to some extent they can be programmed to generate the desired control strategies. For example the head is controlled with a high gain controller while for the arm we employed a low-stiffness control schema. Encoder signals are collected by the same control cards. In some cases also analog data is read directly from the motor control boards.

Images are provided by standard CCD color cameras and they are sampled at full frame rate by frame grabbers sporting the common BT848 chipset. The original images are sub-sampled as early into the processing as possible to the desired resolution and format (within the Babybot always in log-polar format). Auditory signals are sampled at up to 44 KHz by a standard sound card. The signal coming from the microphones is amplified and conditioned appropriately before sampling. Tactile sensors have their own microcontroller and AD converter. Digital values are sent to a PC though a serial line. Hall-effect analog signals are sampled by yet another card with a bank of AD converters.

The hardware is somewhat heterogeneous since it evolved from previous implementation of the Babybot. Control cards have different CPUs, sampling rates, DSP and software interface. The same applies to the set of PCs where the hardware is interfaced to. They range from older Pentium to the latest generation PIV. Presently the robot is controlled by 14 machines connected via two separate 100Mbit Ethernet networks. One network is totally dedicated to control signals, the other mostly to visual processing.

3.3 OS independent software interface

Broadly speaking, the lowest level software components should aim at encapsulating as much as possible the details of the communication and computing layer. The hardware interface should be as seamless integrated into the computation as possible. And, finally, most of the code should easily run on different operating systems depending on the requirements of the specific installation. The language of choice in our case was C++. Development tools are Microsoft Visual Studio for Windows and *gcc* for QNX and Linux.

For the task of encapsulating the operating system we naturally relied on existing software. In particular we found convenient to base our implementation on ACE, an open-source library that among many things provides a tiny object-oriented OS wrapper. For more information about ACE please refer to: http://www.cs.wustl.edu/~schmidt/ACE.html.

ACE runs on Windows, Linux, and QNX that were also our target operating systems. Basing our implementation on ACE allowed clearly running all our code on any of these operating systems. From our point of view ACE provided a common C++ class interface for the communication code and the OS wrapper. Advanced ACE functionalities were not fully exploited. We preferred to take a minimalist approach and rely on the minimum subset of ACE that allowed solving our tasks. Following the open-source philosophy, our software was made freely available on SourceForge:

http://yarp0.sourceforge.net/

The most part of the communication code is profoundly inspired [and recycled] from a previous version developed at MIT (Paul Fitzpatrick. **From First Contact to Close Encounters: A developmentally deep perceptual system for a humanoid robot**. PhD thesis at MIT, 2003) and was tested extensively on the humanoid robot Cog on QNX4.25. The latest implementation has been completely rewritten (using ACE) but it maintains the same high level interface. The distribution is not completely user friendly yet as the only way of obtaining the code is through the CVS interface provided by SourceForge. The library in its entirety has been called YARP (Yet Another Robot Platform).

The communication code is a C++ templatized set of classes contained in a specific static library. The main abstraction for inter-process communication is called a “port”. A port template class can be specialized to send any data type across an IP-network relying on a set of different protocols. Depending on the protocol different behaviors can be obtained – the implemented protocols include TCP, UDP, MCAST, QNET*, and shared memory. A port can either send to many target ports or receive simultaneously from many other ports. A port is an active object: a thread continuously services the port object. Being an active object allows responding to external events at run time, and for example it is possible to send commands to port objects to change their behavior. Commands include connecting to another remote port or receiving an incoming request for connection and since all this can be done at run-time it naturally enables connecting/disconnecting parts of the control system on the fly.

Figure 7 shows an exemplar structure of the port abstraction. Each port is, in practice, a complex object managing many communication channels of the same data type. Each port is potentially both an input and output device although for simplicity of use only one modality is actually allowed in practice. This is enforced by the class definition and the C++ type check. Each communication channel is managed by a “portlet” object within the main port. Different situations are illustrated in Figure 7: for example an MCAST port relies on the protocol itself to send to multiple targets while on the contrary a TCP port has to instantiate multiple portlets to connect to multiple targets. In cases where the code detects that two ports are running on the same machine the IP protocol is replaced by a shared memory connection. In Figure 7 a special portlet is shown: this is indicated as “command receiver”. As already mentioned its function is that of receiving commands to connect, disconnect, or generically operating on the port. Further ports can run independently without blocking the calling process (if desired) or they can wake up the calling process on the occurrence of new data. In some cases synchronous communication is allowed (TCP protocol).

* QNET is the native QNX network protocol (message passing). It is very efficient and tuned for real-time performance.

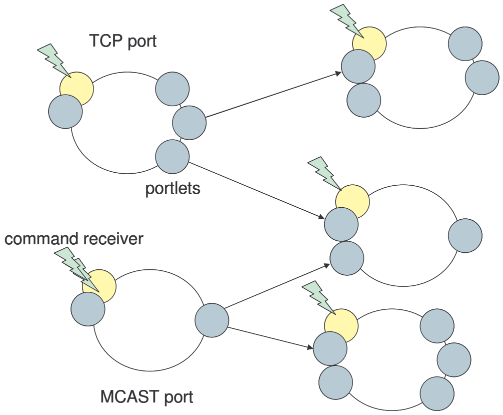


Figure 7: The YARP communication architecture.

Protocols can be intermixed following certain rules. Different operating systems can of course communicate to each other. QNET protocol is an exception and it is only valid within a QNX network.

YARP communication code leads to a componentization of the control architecture into many cooperating modules. The data sent through port can range from simple integral types to complex objects such as arrays of data (images) or vectors. Thus controlling a robot becomes something like writing a distributed network of such modules.
In addition, YARP contains supporting libraries for mathematics and robot type computation (kinematics, matrices, vectors, etc.), image processing (compatible with the Intel IPL library), and general purpose utility classes. We also designed a few modules based on existing Microsoft technology to allow remote controlling Windows machines (this support comes naturally on QNX). In short, these scriptable modules complete seamlessly the architecture allowing the design of scripts to bring up the whole control structure and connect many modules together.

As an aside lately a Matlab interface to ports has been implemented. This allows building Matlab modules (e.g. .m files) that connect to the robot to read/write data. There are basically two advantages: i) complex algorithms can be quickly implemented and tested relying on Matlab existing toolboxes, ii) an additional level of scripting can be realized within Matlab. Matlab provides a relatively efficient and easy to use display library that can be used to visualize the functioning and performance of an ongoing experiment.

In summary, Figure 8 presents schematically the link and dependences between the YARP libraries.

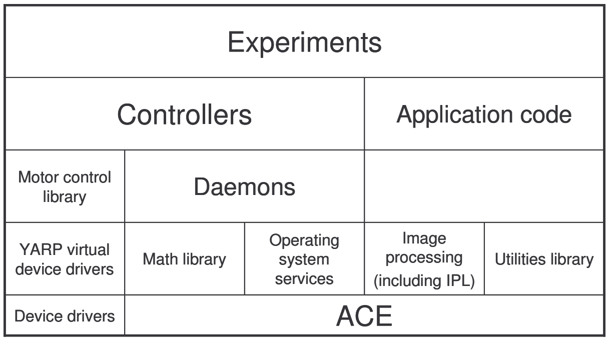


Figure 8 YARP libraries: dependence chart.

3.4 Robot independent code

One of the goals in writing our control architecture has been that of simplifying the programming of a complex robotic structure such as a humanoid robot. As described in the previous section, control cards come in many different flavors and programming them is usually painful. It would be much better if a standardized interface were provided. It would be even better if a suitable abstraction were available.

To solve the first problem we defined a “virtual” device driver interface into YARP. To solve the second, we encapsulated the control of parts of the robot (head, arm, frame grabbers, etc.) into a standardized template class hierarchy.

In short, the virtual device drivers bear much of their structure from the UNIX device drivers. Each card’s driver class contains three main methods: Open, Close, and IOCtl. The latter is the core of the interface. Each device accepts a set of messages (with parameters) through the IOCtl call. Each message accomplishes a specific function. Two different control cards supporting roughly the same commands can be easily (as it was done in our setup) mapped into exactly the same virtual device driver structure, although clearly the implementation might differ.

The next layer is a C++ hierarchy of classes which through templates includes both the specification of the controlling device driver (e.g. the head is controlled through a certain control card) and the idiosyncrasies of the particular setup (e.g. wiring of the robot might differ, or initialization might require different calibration procedures). This hierarchy is shown in Figure 9.

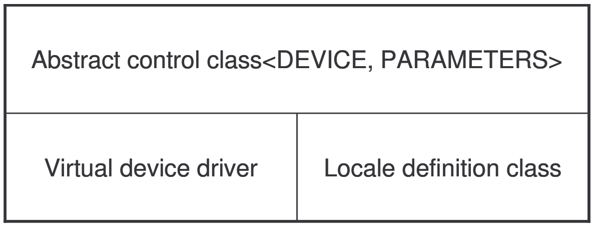


Figure 9: The structure of a control class for a generic device.

3.5 Robot specific interface

The real “communication” with the robot is carried out through a set of binary modules that use the device driver structure described in the previous section. Module customization is at this stage accomplished through configuration files. In the YARP language these modules are called daemons (a term borrowed from UNIX). The daemons directly interact with the remainder of the robot software through YARP ports and in general they export very specialized communication channels. For example the frame grabber has an output port of type “image” and the head control daemon an input port that accepts velocity commands. There are no specific restrictions on the type of ports exported by a daemon since any type of state information about the modules might be required.

Further, some of the daemons accept or send commands of a special type that are generally used to communicate status information. A bus structure based on the MCAST protocol has been implemented to transmit and receive these special messages (called “bottles”). YARP bottles may contain any type of data or even a group of heterogeneous elements of different types. The structure contains identifiers to properly decode messages and interpret the data. YARP bottles create a network within the network of behaviors to realize a high-level control and coordinate a large number of modules.

3.6 Learning architecture†

This layer describes an arrangement of YARP modules that tends to repeat across our robotic architecture. This is not formally into YARP proper but simply an implementation of a particular experiment relying on YARP libraries. Conceptually it forms a layer where to build more sophisticate experiments since for example it provides simple motor control and sensorimotor coordinative behaviors. Overall they could be seen as very high level commands that support positioning, gazing, reaching for visually identified objects, and grasping them.

Grossly speaking, autonomous learning requires a slightly different approach from classical supervised paradigms where data is presegmented and simply fed into a function approximator. Autonomous learning is perhaps closer to reinforcement learning in that it requires action and proper behaviors (exploratory) to gather the training set. Necessarily our architecture will require bootstrapping behaviors supporting building the training set. The question of how much explore and how to get quickly to a solution is an open one in reinforcement learning and unfortunately reinforcement learning itself tend to be difficult, requiring a very large number of samples. In addition, in the case of a real robot we shouldn’t allow “spurious” or random control values to get to the low-level controllers; at the basis of any control strategy we should probably have a reasonable “safe” explorative procedure and certainly not a complete random one. Self-supervised procedures can be identified (similar in spirit to Kawato’s feedback error learning) and given the appropriate amount of exploration they can quickly approximate the desired sensorimotor coordination pattern.

When data samples are available in sufficient number with respect to the size of the parameter space of the function approximator of choice the system can start learning

† Can be skipped on a first read.

and using what has been learnt up to date; necessarily in the long run the influence of explorative behaviors should be reduced. At least two possibilities exist here: learning could be implemented either in batches or fully online. The specific strategy is mostly a function of the algorithm and specific implementation of the function approximation. Inhibition or a functional equivalent should take care of reducing or mixing up exploration with actual "exploitation" of the acquired behavior.

Our discussion is only focused here on the function approximation problem since a good part of the sensorimotor behaviors can be actually well implemented by mapping sensory values onto motor commands or the opposite or even by a combination of the two (e.g. feedback error learning or distal learning). Another constraint on the design of explorative behaviors is that they should mostly "explore" the space that will be used in the future. Needless to say that failure to do so might result in very poor performance.

The learning algorithm can be conceptually divided in two parts: the one providing the "learning signals" sometimes called the "critic", and the one doing the behavior called the "actor". This distinction is important in motor control problems since the actor must be extremely fast and should work in a small delay regime. On the other hand, the critic could take even seconds or minutes to process the training data and provide infrequent adjustments to the actor's parameters. We maintained as much as possible (apart from trivial cases) this distinction within our system. This division is to some extent compatible with biological mechanisms of learning being these for example the rates at which synaptic changes and growth processes develops in the brain compared to actual spikes' travel times.

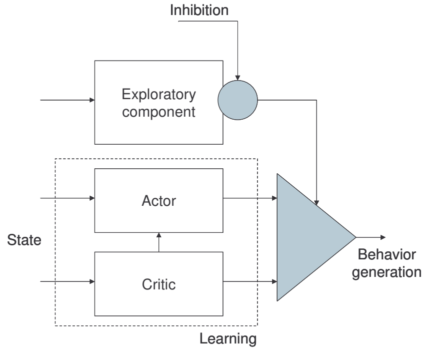


Figure 10: A module for learning sensorimotor coordination.

Figure 10 sketches the modules required for each actual behavior acquisition. At the moment of writing we have only conducted a few experiments with the combination and definition of modules presented here. Examples of explorative components are (at the moment) bounded random behaviors (used when training the hand localization map) or early muscular synergies (simulated muscles) connecting and generating activations of muscles spanning different joints and even different limbs. In learning to reach, these synergies can be exploited to bias the exploration space and avoid random movements. Whenever learning relies on multiple cues, such as visual and motor, having an initial coordination (although imprecise) can be advantageous. One net effect would be the reduction of the learning space that needs to be explored before getting to a reasonable behavior. This strategy was used in our previous work (see G.Metta, G.Sandini and

J.Konczak. *A Developmental Approach to Visually-Guided Reaching in Artificial Systems.* Neural Networks Vol 12 No 10 pp. 1413-1427 (1999)).

The actor and critic modules in our experiment consisted of a simple batch learning backpropagation neural network. Although, not the best, it proved to be very reliable so far. Backpropagation has been extensively tested and its behavior very well characterized in the literature. Consequently, it is much easier to understand especially when things do not go as expected. The implementation maintains the separation of actor and critic to the point of having a slow batch learning method as critic, and a distinct process providing the behavior. Naturally, given the overall robot architecture, the two modules can be even running on two different machines.

Inhibition and the control of activation and coordination of many behaviors is still argument of further research and no definite implementation has been reached yet. Figure 11 shows the combination of many blocks of this type. In this case too, the realization is completely hypothetical since testing has not been performed yet.

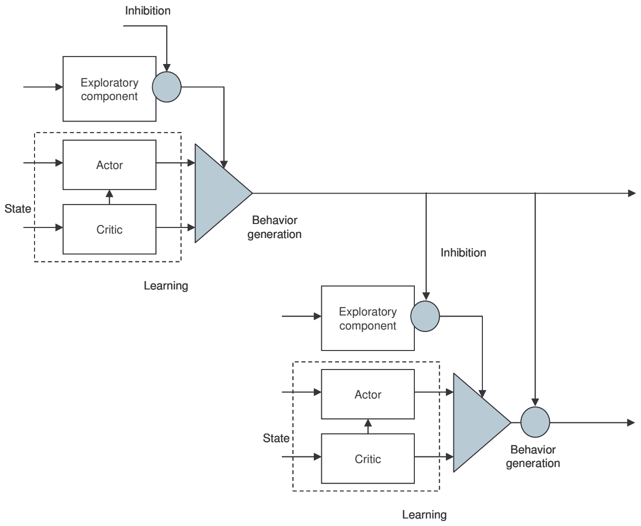


Figure 11: The combination of learning modules in a hypothetical subsumption arrangement.

3.7 James: A Humanoid Robot Acting over an Unstructured World

This platform has been developed after the Babybot following a complete different approach. The electronics and mechanics are custom, specifically designed starting from the requirements of grasping and dexterous manipulation. The robot is described in details in the following paper:

L. Jamone, G. Metta, F. Nori and G. Sandini. *James: A Humanoid Robot Acting over an Unstructured World.* Accepted to the Humanoids 2006 conference, December 4th-6th 2006, Genoa, Italy.
which is available from the RobotCub website. The abstract is reported below:

The recent trend of humanoid robotics research has been deeply influenced by concepts such as embodiment, embodied interaction and emergence. In our view, these concepts, besides shaping the controller, should guide the very design process of the modern humanoid robotic platforms. In this paper, we discuss how these principles have been applied to the design of a humanoid robot called James. James has been designed by considering an object manipulation scenario and by explicitly taking into account embodiment, interaction and the exploitation of smart design solutions. The robot is equipped with moving eyes, neck, arm and hand, and a rich set of sensors, enabling proprioceptive, kinesthetic, tactile and visual sensing. A great deal of effort has been devoted to the design of the hand and touch sensors. Experiments, e.g. tactile object classification, have been performed, to validate the quality of the robot perceptual capabilities.

4 HERTS

4.1 Existing platforms

UNIHER have conducted experiments on two existing robotic platforms: the Sony Aibo robotic dog and the humanoid KASPAR.

Models ERS-220 and ERS-7 Sony Aibo's have been used in the first two years as sensor rich robotic platforms to study construction of sensory-motor maps and deriving sensor-motor laws using information theoretic methods from unknown sensor and actuator configurations.

This embodied robotic research contributed to deliverable D3.2. They have also been used to study interaction histories and metric space of experience, again from an information theory perspective. Robot self-constructed ontologies have been used to play simple games such as peek-a-boo using a simple control architecture that develops as the number of experiences grows.

KASPAR is a humanoid robot developed at UNIHER to carry out research into non-verbal communication, interaction timing and social interaction with minimal facial and manual expressive gestures. Research on KASPAR will also feedback to the RobotCub project on future design of an interactive humanoid robots with regards to their interactive and expressive capabilities.

KASPAR has an 8 DOF head with cameras embedded into the eyes with actuated eyelids. Facial expression is achieved by moving a latex rubber mask at key anchor points to cause opening of the mouth and raising/lowering of the corners of the mouth. Currently KASPAR also has a pair of 3 DOF arms which greatly enhance the gestural capabilities, and will have a pair of 3 DOF humanoid hands to further enhance those capabilities.

4.2 Evaluation

The Sony Aibos were reliable platforms that with their rich sensory feedback both from IR sensors and video camera, as well as from proprioceptive feedback from the motors, supported the development of informational theoretic techniques for embodied robotics.

Experience gained in sensory integration and analysis from working with Aibos is directly applicable to the iCub platform despite the non-humanoid form. Aibos were able to engage in simple interaction games and could potentially engage in fairly complex social interaction. However, lack of manipulation capabilities, poor expressive capabilities, and their dog-like form limits their potential for social interaction.

KASPAR has currently undergone a number of trials with children in classrooms (22 children over 3 weeks) and has proved sufficiently robust for this purpose, although more intensive use will require hardware improvement. Thus far the trials have used Wizard-of-Oz control and software has been developed for this purpose. Sensor feedback currently only consists of the video images but should not prohibit the use of the developed sensor information techniques developed at UNIHER.

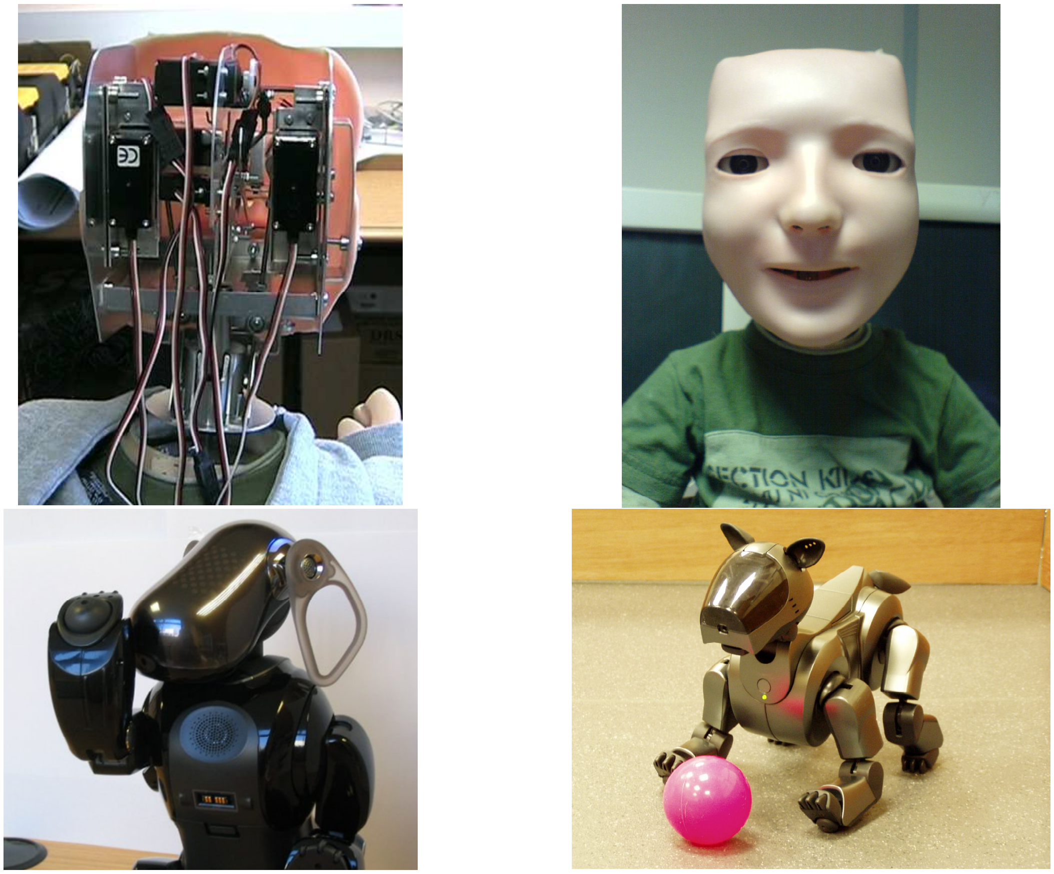


Figure 12: various pictures of the humanoid face Kaspar (upper panel) and the Sony Aibo (lower panel).

5 IST

5.1 The humanoid type Baltazar platform

Baltazar is a humanoid type robot (torso) designed at IST for research in sensory-motor coordination, learning and imitation. The platform consists of a head, arm and hand. The design was driven by the following constraints:

- kinematics similar to that of the human torso. It should be able to perform human-like movements and gestures, as well as to allow a natural interaction with objects (e.g. while grasping).
- Payload of at least 500g (with the hand).
- Force sensing
- Ease of maintenance and "low-cost"

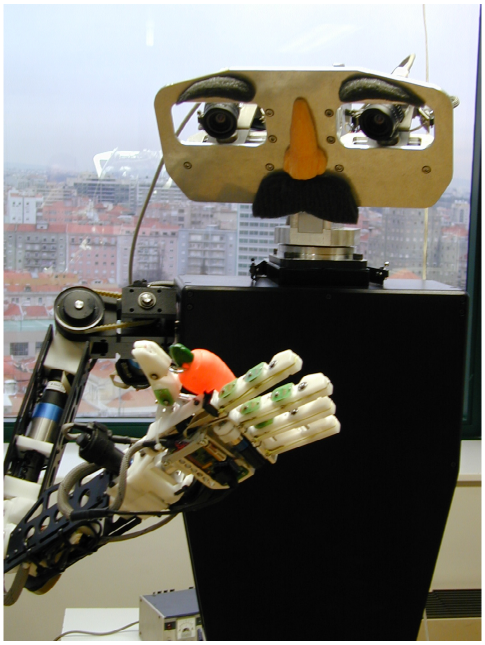


Figure 13: the humanoid robot Baltazar.

5.2 The head

The robot is equipped with a binocular robotic head, previously developed in our lab. It has four degrees of freedom: neck rotation, head elevation and independent eye vergence. Manual adjustments can be made to align the vergence and elevation axes of rotation with the cameras optic centers. The inter-ocular distance can also be modified manually.

5.3 Arm

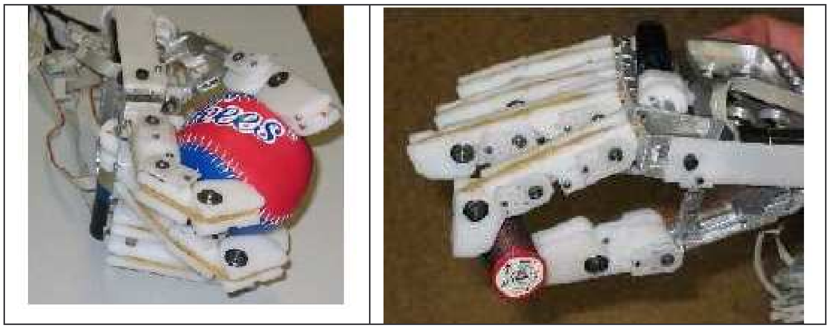
Baltazar's arm has a total number of six degrees of freedom in an anthropomorphic arrangement. The shoulder was modeled with 3 DOF: external/internal rotation, abduction/adduction and extension/flexion. The elbow is endowed with 2 DOF: extension/flexion and supination/pronation. Finally, the wrist possesses 1 DOF: extension/flexion. Wrist abduction/adduction is not included but, to some extent, it is compensated by the hand.

The figure below shows a solution for a vertical working plane (parallel to the robot torso). The different orientations for the hand are obtained by choosing different values for internal/external shoulder rotation.

5.4 Baltazar's hand

The hand has eleven degrees of freedom controlled by four motors included in the hand. The hand can adapt passively to the shape of objects but it can also perform a number of hand gestures that a simpler design would not allow. This design represents a trade off between simplicity and multi-purpose use. We have conducted experiments grasping objects with different shapes: sphere, box, cylinders and general geometric shapes. Force (pressure) sensors have been installed in pulp and palm to better control the exact contact forces.

The index finger has three DOFs. The thumb has two DOFs plus one rotation and the other fingers have two DOFs. The little and ring fingers are mechanically coupled. The abductions were not implemented. The hand is shown in the following figure.



One design constraint was to include all the motors on the hand. For this reason, we could not afford the space to use one independent motor for each joint in the fingers. The index finger is controlled by a single motor that pulls a tendon to close the finger. The thumb has two motors: one for rotation and another for closing the thumb by means of a tendon. The other three fingers have one motor/tendon mechanism that closes them all together.

At a first glance, the fact that not all the DOFs of each finger are controlled independently may seem very limiting. However, the finger joints are also strongly coupled in the human hand. In fact, our design gives some compliance to the robot hand, and makes the grasp control much easier, as the fingers adapt automatically to the shape objects.

The use of a single motor and a tendon to close a finger provides the hand with the ability to passively adapt to the objects shape. However, we need more proprioceptive information (in addition to motor shaft position) to know the state of the hand. The thumb, little and ring fingers have one potentiometer as a position sensor, the middle and index have two potentiometers. In order to interact with objects, we installed pressure sensors in several places of the hand to measure contact forces.

6 EPFL

EPFL currently uses the HOAP2 and HOAP3 humanoid robots from Fujitsu. EPFL-A (headed by Aude Billard) uses the HOAP 3 for upper limb movement control (e.g. manipulation tasks) and research on learning by imitation, as part of WP3, WP5 and WP6. EPFL-B (headed by Auke Ijspeert) uses the HOAP 2 for locomotion control (biped locomotion) and for testing control mechanisms based on dynamical systems with well-defined attractor properties, as part of WP3.

The HOAP 2 specifications are the following. The robot has a height of 50cm and a weight of 7 Kg. It has 25 degrees of freedom (6 in each leg, 5 in each arm, 1 in the torso, 2 in the head). Except for the head two DOFs which are controlled by servos, all DOFS are driven by DC motors with their own gearboxes and control loops. The robot can be controlled by an onboard CPU or an offboard PC running Real-Time Linux with a 1ms control cycle. The communication protocol is USB 1.0.

The HOAP 3 is very similar to the HOAP 2. It is slightly larger and heavier (60cm and 8.8Kg). The kinematic structure is close to that of the HOAP 2 with the addition of one DOF in each arm (wrist rotation). It has been extended with two cameras, a microphone, a speaker, expression LEDs, audio recognition function, and voice synthesis function.

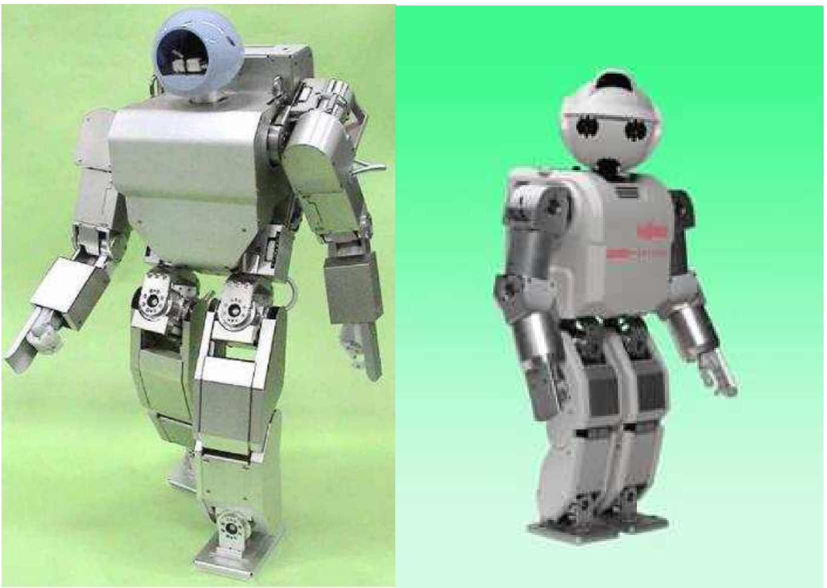


Figure 14: humanoid robots HOAP 2 (left) and HOAP 3 (right) from Fujitsu currently used at the EPFL.

The pros of these platforms are the following:

1. Easy to work with. These robots are sold for research, and therefore provide access to low-level control and details for the interfaces with the hardware.
2. Robust design and good packaging. As a commercial product which has now gone through three iterations, the robots are well designed and suitable for extensive experiments.
3. Good size and weight. The robots are easy to manipulate and do not need a large lab for carrying experiments. A single person is sufficient to manipulate them, and there are no particular safety issues to follow when carrying out experiments.
4. Real-time Linux. The robots use an open-source OS with all the advantages that it brings (access to the source codes, access to a community of enthusiasts, possibility to modify the OS).

The cons of the platforms are the following:

1. Very simplified hands. The robots are not well suited to carry out fine manipulation tasks because their hands have all fingers actuated by a single motor through tendons for grasping objects.
2. Not well suited for crawling. The robots are designed for biped locomotion, and are not well suited for testing crawling controllers like those currently developed for the iCub. There is a risk of damaging parts of the robot because the knees and hands are not designed to sustain impacts and carry large loads.
3. The robots are mainly usable with a tether. There is the option to run the robots without tethers but the CPUs on board have limited computing capabilities, and the batteries on board allow only limited autonomy (less than 30 minutes).
4. Relatively slow movements and limited accelerations.
5. Poor documentation, at least in English. Most customers are Japanese labs and the English documentation could be improved.
6. Only relative joint angle position sensors (incremental encoders). The robots need to be recalibrated each time it is turned off or rebooted. Although the manufacturer claims that it is possible to do torque control, this feature is not documented and it was not possible to activate it at this stage.

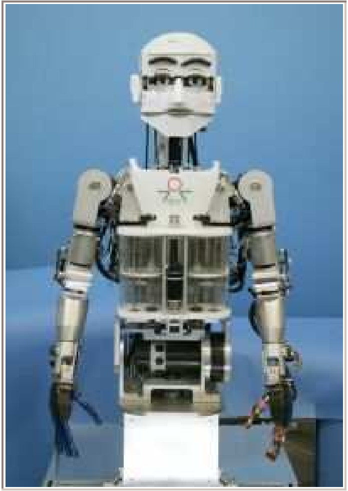
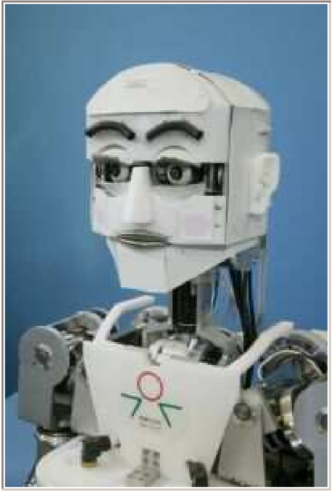
The iCub will fulfill all of the cons above, while providing points 1 and 4 of the above-pros. The consortium will have to make sure that points 2 and 3 of the above pros of the HOAP robots will be pros of iCub as well. The three stage testing is effected in RobotCub with duplication of the iCub at three places in the consortium and it will play a key role in ensuring the user-friendliness of the robot’s use. Safety and ease of use, as stated in point 3, will, however, be key issues for the use of the robot as part of human-robot interaction and deserve particular attention during the realization of the prototype.

7 SSSA

7.1 WE-4RII

Since 1995, Emotion Expression Humanoid Robots have been developed at Waseda University. The goal is the development of new mechanisms and functions for a humanoid robot having the ability to communicate naturally with a human by expressing human-like emotion. In 2003, 9-DOFs Emotion Expression Humanoid Arms were developed to improve the emotional expression. The arms were integrated with WE-4 (Waseda Eye No.4) to develop the Emotion Expression Humanoid Robot WE-4R (Waseda Eyes No.4 Refined) that could express its emotions by using its facial expressions, torso, and arms. In 2004, researchers of Scuola Superiore Sant'Anna and Waseda University developed the WE-4RII (Waseda Eye No.4 Refined II) by integrating the anthropomorphic robot hand RCH-1 (RoboCasa Hand No. 1) to WE-4R. RCH-1 has 6-DOFs and abilities for emotion expression, grasping and tactile sensing. This robot is in Tokyo at Robocasa (SSSA/Waseda joint lab).

The following figures show the hardware of the Emotion Expression Humanoid Robot WE-4RII. It has 59-DOFs (Hands:12, Arms:18, Waist: 2, Neck: 4, Eyeballs: 3, Eyelids: 6, Eyebrows: 8, Lips: 4, Jaw: 1, Lungs: 1) and a lot of sensors which serve as sense organs (Visual, Auditory, Cutaneous and Olfactory sensation) for extrinsic stimuli.

	
WE-4RII (Whole View)	WE-4RII (Head Part)

The eyeballs have 1-DOF for the pitch axis and 2-DOF for the yaw axis. The maximum angular velocity of eyeballs is similar to a human with 600[deg/s] for the eyeballs. The eyelids have 6-DOF. WE-4RII can rotate its upper eyelid in order to be able to express using the corner of robot's eye. The maximum angular velocity of opening and closing eyelids is similar to a human with 900[deg/s] for the eyelids. Furthermore, this robot can blink within 0.3[s], which is as fast as a human does.

For miniaturization of the head part, we newly developed an Eye Unit that integrated eyeballs parts and eyelids parts. Moreover, in the Eye Unit of WE-4RII, the eyeball pitch axis motion mechanically synchronizes opening and closing upper eyelid motion. Therefore, we can control coordinated eyeball-eyelids motion by hardware.
WE-4RII's neck has 4-DOF, which are the upper pitch, the lower pitch, the roll and the yaw axis. WE-4RII can stretch and pull its neck using the upper and lower DOF like a human. The maximum angular velocity of each axis is similar to a human's at 160[deg/s].

WE-4RII has 9-DOFs Emotion Expression Humanoid Arms. The arm consists of a base shoulder part (pitch and yaw axis), a shoulder part (pitch, yaw and roll axis), an elbow part (pitch axis), and a wrist part (pitch, yaw and roll axis). By using the 2-DOFs of the base shoulder, it can move the whole shoulder up and down, back and forth. This enables WE-4RII to do such movements as squaring its shoulders when angry or shrugging its shoulders when sad. Therefore, WE-4RII can express its emotions effectively by using its arms.

As for a hand part, development was performed at the joint laboratory for reserch on humanoid & personal robotics ROBOCASA. Anthropomorphic robot hand RCH-1 (RoboCasa Hand No.1) was designed and developed for WE-4RII at SSSA (Scuola Superiore Sant' Anna) ARTS Lab in Italy.

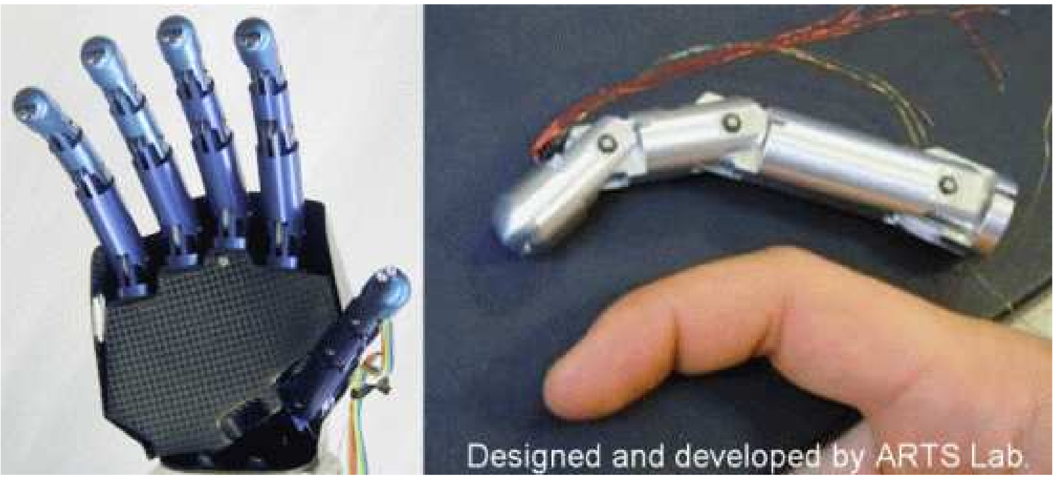


Figure 15: RCH-1.

RCH-1 has three tactile sensors; distributed on/off contact sensor, FSR, 3D force sensor. Distributed on/off contact sensor is the switch consisted of thin sheets. It is arranged at 16 places; 15 parts for inside of the finger, 1 part for palm. The FSR is the same used for the head part of WE-4. By using two layered sheet, we can recognize not only the magnitude of the force, but also the difference of the touching manner that are "Push", "Stroke", "Hit", and 3D force sensor which can measure the fingertip force is implemented in the fingertips of the thumb and index finger.

WE-4RII has 2-DOF waist composed by pitch and yaw axes. By using the waist motion, WE-4RII can produce emotional expression with not only neck but also the upper-half part of its body.

WE-4RII expresses its facial expression using its eyebrows, lips, jaw, facial color and voice. The eyebrows consist of flexible sponges, and each eyebrow has 4-DOF.
We used spindle-shaped springs for WE-4RII's lips. The lips change their shape by pulling from 4 directions, and WE-4RII's jaw that has 1-DOF opens and closes the lips. For facial color, we used red and blue EL (Electro Luminescence) sheets applied on the cheeks. WE-4RII can express red and pale facial colors.

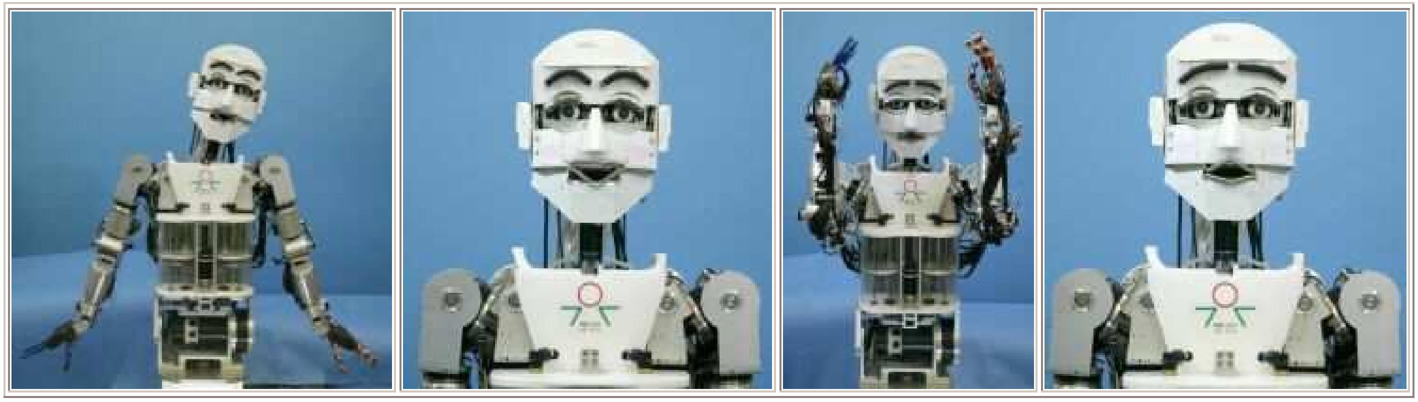
For the voice system, a small speaker was set in the jaw. The robot voice is a synthetic voice made by LaLaVoice 2001 (TOSHIBA Corporation).

WE-4RII has two color CCD cameras in its eyes. The images from its eyes are captured to a PC by an image capture board. WE-4RII can recognize any color as the targets and it can recognize eight targets at the same time. After calculating the gravity and area of the targets, WE-4RII can follow them with the eye, the neck and the waist. This makes it possible to follow the target with any color in the three dimension space.

Two small condenser microphones are used as auditory sensation. WE-4RII can localize the sound directions from the loudness between the right and the left.
WE-4RII has tactile and temperature sensations in the human cutaneous sensation. FSR (Force Sensing Resistor) are used as tactile sensation as it is able to detect even very weak forces, and is a thin and light device. A method for recognizing not only the magnitude of the force, but also the difference of the touching manner that are "Push", "Stroke", "Hit", has been implemented by using a 2 layers structure with FSR. On the other hand, WE-4RII has a Thermistor the temperature sensor. FSRs are also installed on the palms to detect whether it has been contacted or not.

Four semiconductor gas sensors are embedded in WE-4RII's nose for replicating the olfactory sensation. WE-4RII can recognize the smells of alcohol, ammonia and cigarette smoke.

We use the Six Basic Facial Expressions of Ekman in the robot's facial control, and have defined the seven facial patterns of "Happiness", "Anger", "Disgust", "Fear", "Sadness", "Surprise", and "Neutral" emotional expressions. The strength of each emotional expression is variable by a fifty-grade proportional interpolation of the differences in location from the "Neutral" emotional expression. The speed of the arm movement is changed according to the emotion of the robot. Therefore, the emotion of the robot can be expressed by both the posture and the speed of the arms. WE-4RII has the emotional expression patterns shown in the following figures.



WE-4RII changes its mental state according to the external and internal stimuli, and expresses its emotion using facial expressions, facial color and body movement. There are two big information flows in the robot. The one is the flow caused from the external environment. And, the other is the flow caused from the robot internal state.

Furthermore, we introduced the Robot Personality because each human has deferent personality. The Robot Personality consists of the Sensing Personality and the Expression Personality. The need and the emotion are a two-layered structure, and the need is in a lower layer than the emotion because we thought that the need was nearer to the instinct than the emotion. Furthermore, the need and emotion affect each other through the Sensing Personality.

The Robot Personality consists of the Sensing Personality and the Expression Personality. The former determines how a stimulus works the mental state. And, the later determines how the robot expresses its emotion. These personalities can be easily assigned. Therefore, it's possible to easily obtain a wide variety of the Robot Personalities. Moreover, we introduced the "Learning System" in order for the robot to learn the experiences and construct its personality based on its experiences dynamically.

As the iCub is still being developed it is hard to compare it to WE-4RII on the actual skills. The WE-4RII is provided with lots of sensors and it is reliable and robust; this aspect is extremely important for interacting properly with humans. Moreover facial expressions are implemented in this platform and not in the iCub.

Anyway the WE-4RII is not a full-bodied platform. The mobility will provide the iCub with a wider set of capabilities (i.e. the exploration of the environment) and will make it really an open platform. Last but not least, the iCub hand is the evolution of the RCH-1. Now 20 DoFs (with 8 motors) are implemented instead of 16 DoFs (with 6 motors); hence a wider set of grasp typologies and dextrous manipulation are available.

7.2 ARTS Humanoid Platform

The ARTS humanoid platform is a 1-link trunk that supports one arm-hand system and a neck-head system. The 2-dof trunk is part of the arm (Dexter arm, by S.M. Scienzia Machinale srl, Pisa, Italy) which has in total 8 dofs, and integrates the 4 motors of the three-fingered hand on the forearm. The hand has anthropomorphic dimensions and weight. Each finger consists of 3 underactuated dofs driven by a single cable allowing flexion/extension. A 2-dof trapezo-metacarpal joint at the base of the palm allows thumb opposition movement (adduction/abduction). In total, the hand has 10 dofs, 6 of which are underactuated. The four hand motors are located on the forearm. The perception system of the hand includes proprioceptive and exteroceptive sensory systems. Each finger is equipped with three position Hall-effect sensors (one per phalanx), one 3D force sensor embedded in the fingertip, 3 on/off contact sensors (one per phalanx). In addition, the four hand motors are equipped with encoders.

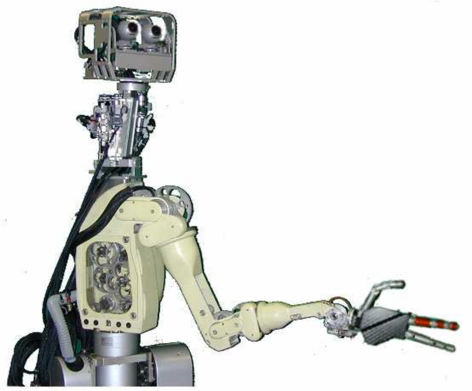


Figure 16: the ARTS humanoid platform.

The anthropomorphic robotic head was designed with reference to the physical structure and performance of the human head regarding dofs, ranges of motion, speeds and accelerations. Thus, the head has seven dofs equipped with incremental encoders for measuring the positions of all the joints as proprioceptive information: 4 dofs on the neck (1 yaw, 2 pitches at different heights, 1 roll), 1 dof for a common eye tilt movement and 2 dofs for independent eye pan movements. The head was equipped with two cameras.

7.3 Hand mechanical specifications

10 dofs; 6 underactuated, 4 motor actuated, three identical underactuated 3 dof fingers with cylindrical phalanges, driven by a single cable allowing flexion/extension
a 2 DoF trapezo-metacarpal joint at the base of the palm allowing thumb opposition movement (adduction/abduction) towards the other 2 fingers.

Weight: about 400gr
Dimension: similar to the human hand

This platform is aimed to link action and perception through vision, proprioception and exteroception. The results obtained during years of tests on this platform will be finally exploited in an advanced full-humanoid robot, the iCub.

Reference: P. Dario, M.C. Carrozza, E. Guglielmelli, C. Laschi, A. Menciassi, S. Micera, F. Vecchi, *Robotics as a Future and Emerging Technology: biomimetics, cybernetics and neuro-robotics in European projects?*. IEEE Robotics and Automation Magazine, Vol.12, No.2, June 2005, pp.29-43.

7.4 CYBERHAND

The ultimate goal of the project related to this hand is the development of a bio-inspired hand prosthesis control system for performing experiments of efferent motor control, based on electroneurographic (ENG) signals recorded by a neural interface. These efferent nervous signals, classified by a software (ENG classifier), will be able to drive the hand actuators. Hence a great effort is addressed in developing a sensory system, mimicking the natural mechanoreceptors in the human hand. Consequently the prosthesis employs several sensors such as cable tension sensors, joint angular hall

sensors, finger global flexion sensors (motor encoder), motor stroke end sensors, on-off tactile sensors and fingertips three-axial force sensors.
Nevertheless the CYBERHAND comes from the previous robotics hands 'Paloma' and 'RCH-1' as the bio-inspired approach of the design and the sensory systems are the same. Furthermore the results obtained have already been exploited back to the iCub and other robotic hands.

The hand is actuated by six DC motors, one for each finger (cable driving) plus one for the thumb opposition. The thumb opposition motor is inside the palm, while the motors for the flexion of the fingers are all located outside on a mechanical platform.
The underactuated system of the prosthesis allows the flexion of the three joints of the finger with a single DC motor. Although this characteristic does not permit to control independently each phalanx, each finger configuration is determined by the shape of the object to grasp (that acts as external constraint), making the grasp autoadaptive. This is achieved by the means of the connected differential mechanisms. Hence it is possible to control a large number of DoFs with a limited number of actuators. The goal indeed is to obtain a stable grasp with objects of different size and shape, without increasing the complexity of the mechanism and of the control algorithm. This feature is particularly important in prosthetic hands where the constraints are:

- weight and dimension;
- low power consumption;
- low signals availability.

An experimental platform has been developed with a shape that mimics the human forearm.

The tendon tensiometer is based on strain gauges sensors (model ESU-025-1000, Entran Device Inc, Fairfield, NJ, USA). The micromechanical structure has been fabricated to obtain a cantilever (see figure below) elastically strained by the cable, in order to continuously monitor the cable tension applied by the motors, similarly as the Golgi tendon organ in series with a muscle.

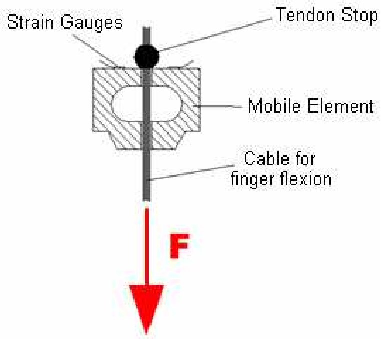


Figure 17: cable tension sensor.

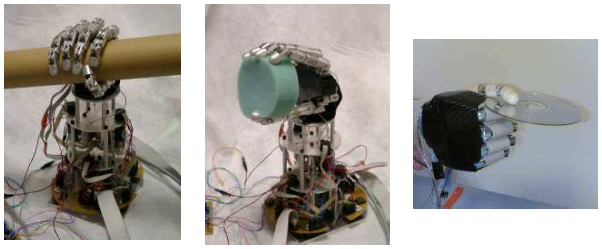
In order to obtain both high sensitivity and mechanical strength, a FEM analysis has been performed, using width, thickness, radii of the cantilever as parameters (material is defined by the tension and the overall dimension: C40 steel). The experimental results show a good repeatability and linearity, and a maximum hysteresis error of about 8% of full scale.

The open platform presents several electronic resources and numerous sensors, i.e. five cable tension sensors (one for each finger), Hall effect sensors (one for each joint) for phalanxes position measurement, motor encoders, motor stroke end sensors, on-off tactile sensors and three fingertips three-axial force sensors.

The controller developed exploits the signals provided by the twelve stroke end sensors and the five cable tension sensors. The output signals of these five cable tension sensors are proportional to the grip force applied by each finger during the grasp of an object. Because of the impossibility of controlling each degree of freedom of an underactuated mechanism, precision prismatic grasps were not involved in the experiments. For this reason, in the experiments “precision” grasps have been substituted with “light” grasps. The difference between power and light grasps is only in the grip force value fixed in the control algorithm; moreover, the grasp type is selected during the experiments according to the object weight.

The control algorithm is composed of two subsequent and different phases. After the first phase of fingers pre-shaping (performed by PID motion controllers), desired fingers tendons force is calculated according to the measured cable tension and to the grasp class. In the second phase, the hand closes the involved fingers until the desired grasp force is reached. The obtained grip performs a bio-inspired balanced distribution of the forces within the hand and each finger grips the object with the same force. A balanced distribution of the forces is desired for several reasons: first because this is what happens in the human hand. The artificial hand is an underactuated device capable of performing auto-adaptive grasps; hence, it is preferable to control the global tight and not each finger separately. Moreover, the mechanical stress produced by the grasp is equally distributed on each part of the hand (fingers mechanical structure, cables, cable sensors) reducing the risk of damage. Furthermore, grasp stability is increased and slippage risk is reduced.

Figure 18 shows several grasps performed with high stability as a result of the implemented control strategy.



(a) Small Diameter

(b) Round sponge

(c) CD

Figure 18: grasping capabilities of the hand.

The hand is able to perform rapidly, stable grips in all cases shown in Figure 18. Furthermore, experiments have proved that precision grasps are hardly obtainable due to the mechanical characteristics of the hand. The control strategy used in this simple but robust controller allowed performing stable grasps in the 96% of the experiments. Concerning the mechanical design and the sensory system, the Cyberhand can be defined as ‘mother’ of the SSSA iCub hand. Obviously there are differences. First of all the Cyberhand is shaped and dimensioned as an adult man’s real hand, hence is more robust and reliable than the iCub hand. Nevertheless the latter one is provided with a similar proprioceptive sensory system and with 20 DoFs instead of 16.

8 UNISAL

8.1 Pneumatic Muscle Actution (pMA)

For robotic and general mechatronic applications electric motors (either ac or dc) form the most common choice, while for outside tasks involving construction and agriculture hydraulics have many benefits. Each of these systems have been very successful but they possess well known and documented limitations when applied to robotic applications, particular those involving interaction with people - a scenario increasingly seen as important for future generations of robots Research into new actuation systems to attempt to address the concerns raised by conventional electric and hydraulic designs have inspired the development of mechanisms such as polymeric actuators, ultrasonic motors, Shape Memory Metals (SMM), magneto-strictive actuators, Electro-rheological fluids. Among the most promising of these new actuation systems are pneumatic Muscle Actuators. These pneumatic Muscle Actuators (pMA) are constructed as a two-layered cylinder. This design has an inner rubber liner, an outer containment layer of braided nylon and endcaps that seal the open ends of the muscle.

The structure of the muscles gives the actuator a number of desirable characteristics:

1. This muscle can be made in a range of lengths and diameters with increases in sizes producing increased contractile force.
2. Actuators have exceptionally high power and force to weight/volume ratios.
3. The actual achievable displacement (contraction) is dependent on the construction and loading but is typical 30% of the dilated length - this is comparable with the contraction achievable with natural muscle.
4. Being pneumatic in nature the muscles are highly flexible, soft in contact and have excellent safety potential. This gives a soft actuator option which is again comparable with natural muscle.
5. Controllers developed for the muscle systems have shown them to be controllable to an accuracy of better than 1% of displacement. Bandwidths for antagonistic pairs of muscles of up to 5Hz can be achieved. Force control using antagonistic pairs of muscle (compare with muscle action) is also possible.
6. When compared directly with human muscle the contractile force for a given cross- sectional area of actuator can be over 300N/ cm2 for the pMA compared to 20-40N/cm2 for natural muscle.
7. The actuators can operate safely in aquatic or other liquid environments and are safe in explosive/gaseous states.

8.2 Salford pMA Biped-I

Work on the development of a lightweight pMA humanoid robot started in Salford University in 1996. The aim was to investigate the actuation potential of the pMA. The robot design was kept relatively simple [18-20]. The robot, Figure 19 was mainly constructed from aluminium. The total mass was less than 5kg (no valves or control structures were on board) and the height was about 1.15m from the ground to the hip joint. The original model of the biped had 4 active joints per leg; an inverted pendulum spine, hips, knees, and ankles [1-3].

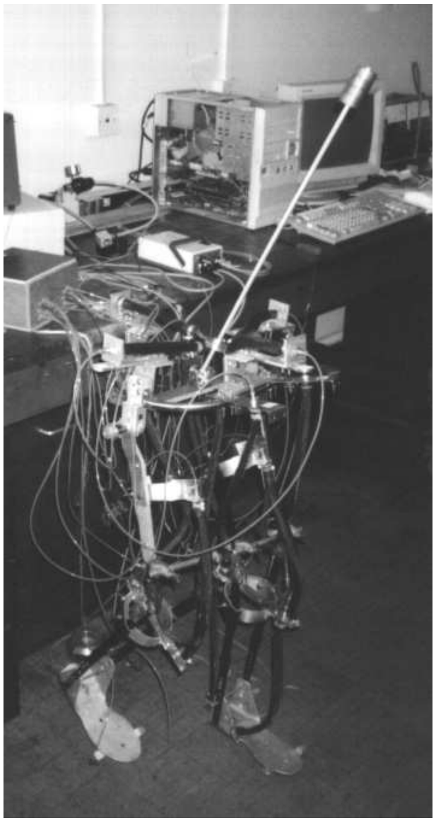
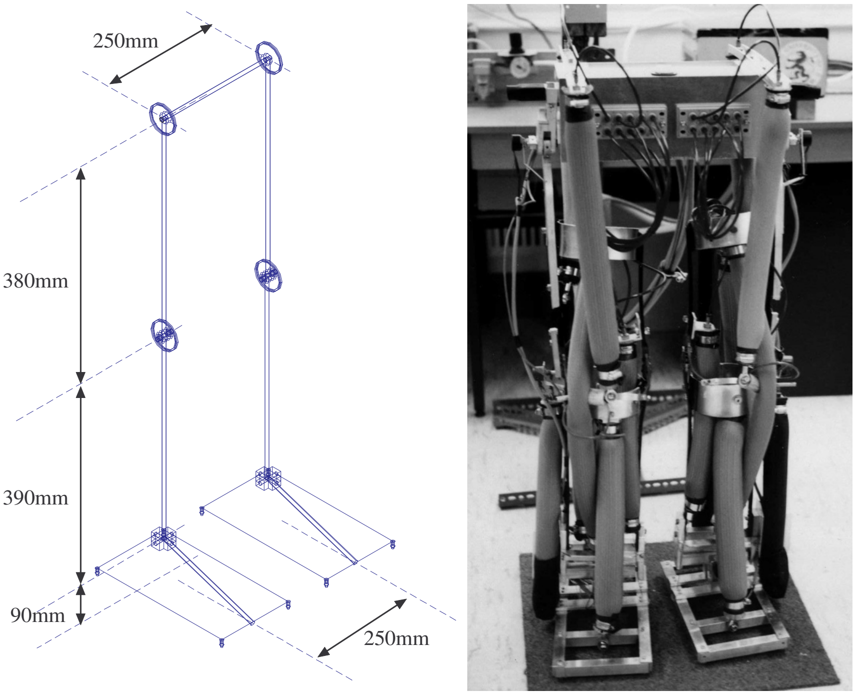


Figure 19: First prototype pMA Biped in 1997.

This first prototype pMA biped, had an aluminum spinal column assisting the transition of the Centre of Gravity (COG). Two antagonistic pairs of pMA muscles were employed to manipulate the spinal column in frontal and saggital planes. This robot achieved limited stepping motions determined by the swing of the spinal column.

8.3 pMA Biped-II

This early work had shown that the pMA muscle could provide an adequate actuation system for a very simple mechanical biped. However, it was realized that better performance could be achieved with modification to the construction and electronic hardware. Therefore the pMA Biped-I was modified and developed to form the pMA Biped-II, Figure 20.



a) b)

Figure 20: The pMA Biped-II, a) dimension and b) actual robot.

The major modifications of the pMA Biped-II compared to the initial design are at ankles, feet, and pelvis. The pMA Biped-II has 5DOF each leg, 2DOF in each ankle, 1DOF in each knee, and 2DOF in each hip. The modifications to form the pMA Biped-II produced a robot with a total height when standing of 860mm from the ground to the hip joint. The height from the ground to the ankle joint was 90mm, the height from the ankle joint to knee joint was 390mm, and from the knee to the hip joint was 380mm. There is an additional 100mm from the hip to the top of pelvis box giving a total height of 960mm. The mass of the biped was 13kg without the valve system but including all the actuators. The pMAs that powered every joints of pMA Biped-II had a diameter of 40mm. This size of pMA muscle gave a maximum contraction force of 500N/bar.

8.4 pMA Biped-III

The experiences of the previous pMA bipeds suggested that the kinematics model and construction needed to be enhanced if dynamic walking and active balance were to be achieved reliably. Therefore, a new biped (pMA Biped-III) was designed, developed and constructed based on the experience obtained from previous generations. The pMA Biped-III is mainly constructed in GRP (Glass Reinforced Polyester) composite and aluminum to minimize the mass of the structure but also because the use of composite has a direct analogy to human bone and can form a sound basis for ‘soft’ robotic structures. Steel is employed for the high-load sections.

The pMA Biped-III Figure 22 has been designed to provide weight distribution roughly at the middle of each link to aid in modeling and control. All pMA muscles on the robot have the same length and diameter, which are 315mm (275mm active length) and 40mm, respectively. This size of muscle can produce a maximum force of 500N/bar and a maximum contractile displacement of 96.25mm (35% contraction). At each joint, a stainless steel shaft is employed together with dry bushes bearings to reduce the friction. The pMA Biped-III has 10 DOFs in total, 2DOFs in each hip, 1DOF in each knee, and 2DOFs in each ankle. The standing height of the pMA Biped-III from the ground to the hip flexion/extension joint is 1028mm. There is an additional 125mm from the hip to the top of pelvic box giving a total height of 1153mm. The total mass of the pMA Biped-III is approximately 14kg.

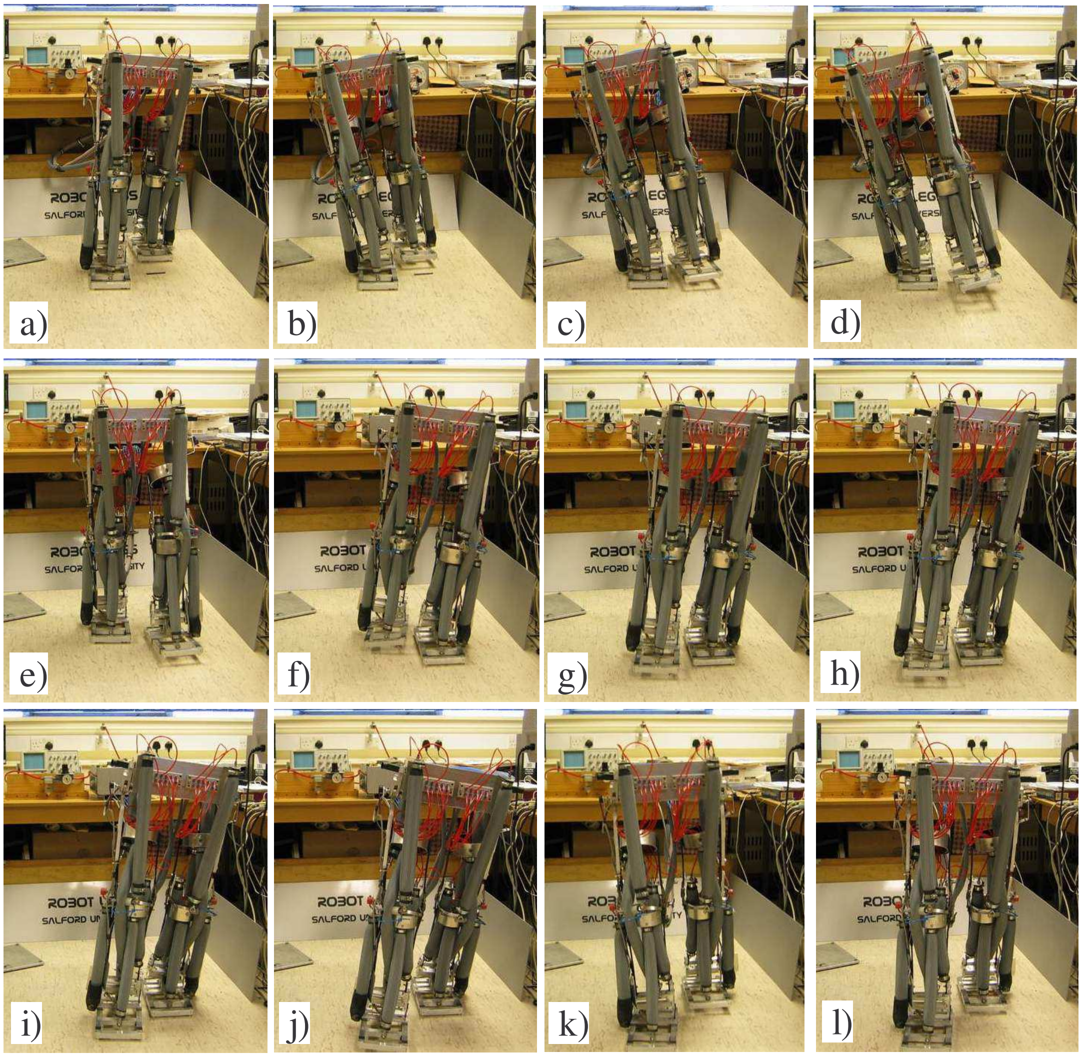


Figure 21: the pMA Biped-II full gait walking, a) the initial position, b) weight transfer to the right foot becomes single support phase, c) to e) left leg swing, f) weight transfer from the right foot to the left foot to become single support phase, g) to j) right, and finally stabilizing k) and l).

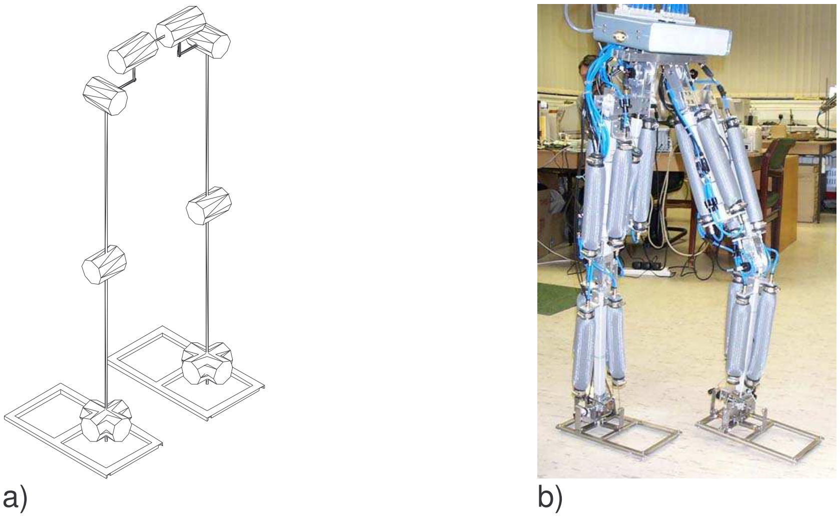


Figure 22: pMA Biped-III.

With this system the step pattern can produce steps of up to 0.35m in periods as short as 0.25s. For such large steps and such rapid movement the present controller is unable to consistently ensure stability following the fall transition and for repeatability the step size is currently limited to a demand of 0.25m and a step time of 0.5-1s. This is mainly due to the low control bandwidth that is the result of the air flow rate limitations as set by the pneumatic circuit (valves, size of pipes, etc)

This work showed that the application of the pMA in robotic systems is feasible despite the major drawbacks due to the accuracy and the difficulty of control. Despite these issues and the high tolerance misalignment of the pMA Biped-II, it succeeded in walking and this gave a confident to the pMA system to be integrated in the new approach to the robotics technology - the biological inspired- such as the humanoid robot.

8.5 References

[1] D. G. Caldwell, G. A. Medrano-Cerda, and C. J. Bowler, "Investigation of Bipedal Robot Locomotion using Pneumatic Muscle Actuators," presented at Proceedings of the 1997 IEEE International Conference on Robotics & Automation (ICRA1997), New Mexico, 1997.

[2] D. G. Caldwell, N. Tsagarakis, D. Badihi, and G. A. Medrano-Cerda, "Pneumatic Muscle Actuator Technology a light weight power system for a Humanoid Robot," presented at Proceedings of the 1998 IEEE International Conference on Robotics & Automation (ICRA1998), Leuven, Belgium, 1998.

[3] D. G. Caldwell, N. Tsagarakis, W.S.Yin, and G. A. Medrano-Cerda, "'Soft' Actuators-Bio-mimetic Systems for a Bipedal Robot," presented at International Conference on Climbing and Walking Robots (CLAWAR 1998), Brussels, 1998.

[4] P.Artrit, N.G.Tsagarakis, G.A.Medrano-Cerda and D.G.Caldwell, “Biomimetic Humanoid Robot: Biped Locomotion Biomimetic Humanoid Robot: Biped Locomotion”, CLAWAR 2000, Madrid September 2000.

9 UNIZH

9.1 Hardware and software platforms at the AILab

Our robotic setup consists in the following components: a robotic hand, an active color stereo vision system, a conventional robot arm, a robot arm with artificial muscles.

9.2 Robotic hand

Our current prosthetic hand prototype has 13 DOF in total and consists of four fingers with 3 joints and 2 DOF each, one thumb with 2 joints and 3 DOF, and two additional DOF for the wrist. All the joints are actuated by tendon wires directly connected to a pair of servomotors, providing 2 DOF using the adaptive joint mechanism developed by (Ishikawa et al., 1999). The pressure sensors are placed on the fingertips, on the base of each finger and in the palm. Each finger has a flex/bend sensor and an angle sensor. The total weight of our current prototype is 1.2 Kg (including motors and tendons). A detailed model of the robotic hand (i.e., kinematics, position of sensors, pressure sensors) has been programmed within a realistic physics environment based in the open dynamics engine library (ODE) (see Figure 23c and Figure 23d).

9.3 Stereo color active vision system

A six DOF robotic head, each camera can be panned and tilted independently, the two additional DOF are for actuating the neck. The images from the cameras are recorded at a rate of 30 frames per second and a resolution of 320x240 (see Figure 23b and Figure 23e). Algorithms for visual processing (e.g., color detection, motion detection based on the optical flow, direction detection, log polar transformation, etc.) are already in place.

9.4 Conventional robot arm

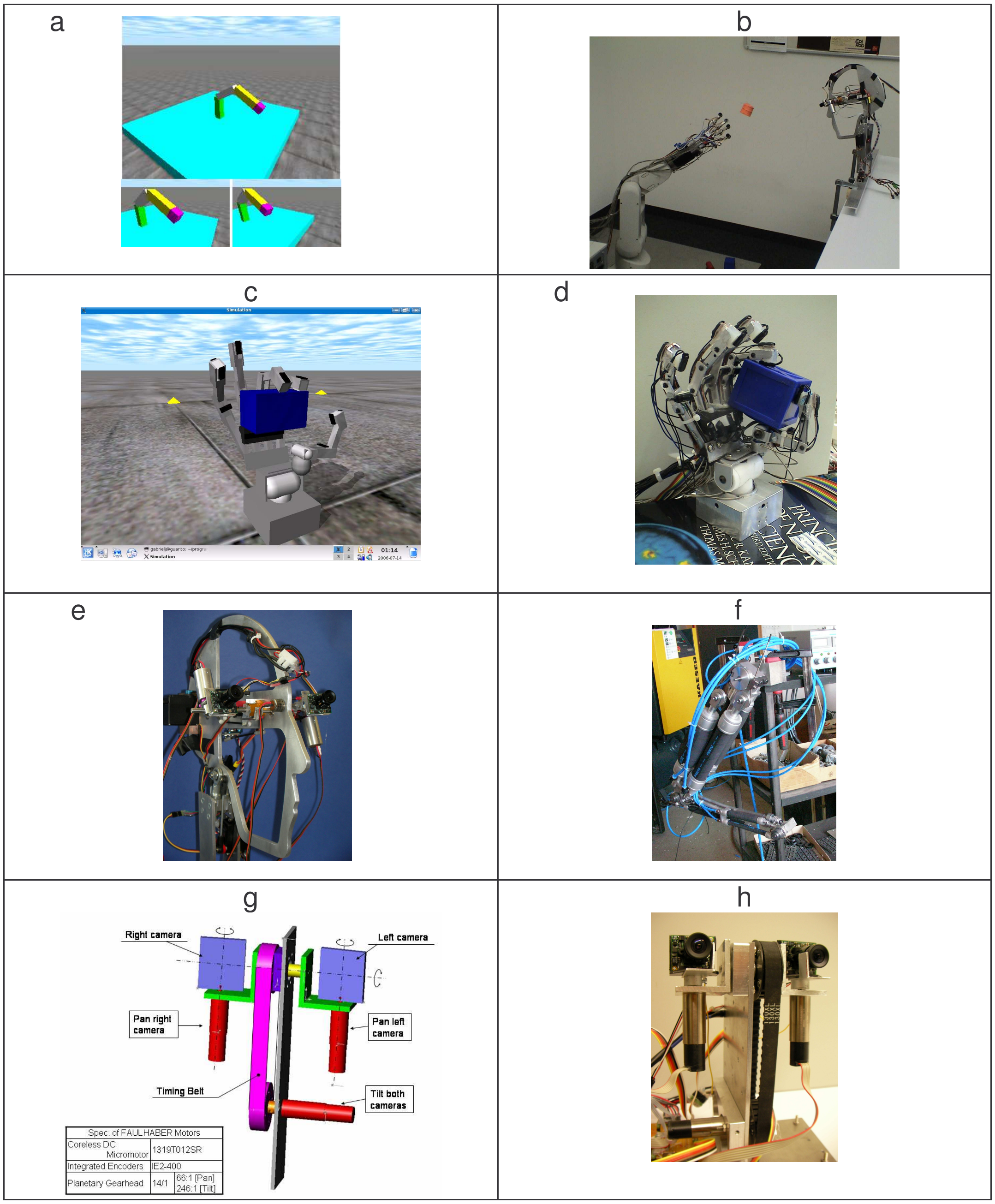
An industrial robot manipulator (Mitsubishi® MELFA RV-2AJ) with five degrees of freedom (i.e., shoulder, elbow, and wrist) and a special connector to hold the robotic hand as illustrated in Figure 23b.

9.5 Robotic arm with artificial muscles

Pneumatic systems offer great benefits concerning speed, weight and force. We have made some preliminary experiments to study ways in which the non-linear dynamics, redundancy, and compliance of the system can be exploited to achieve more natural kinds of movements. A first prototype built from FESTO pneumatic actuators is already available and an improved version is under construction (Figure 23f).

9.6 iCub's active vision system

We designed and manufactured a test bed for an active vision system, based on the iCub's design using the same motors and cameras (Figure 23g and Figure 23h).



Spec. of FAULHABER Motors		
Coreless DC Micromotor	1319T012SR	
Integrated Encoders	IE2-400	
Planetary Gearhead	14/1	66:1 [Pan] 246:1 [Tilt]

Figure 23: Illustrations of the different simulators and hardware platforms. (a) ODE simulation of the robot arm and the active vision system. The lower two images are from the perspective of the two cameras observing the robot arm (b) Complete robotic setup showing the active vision system, as well as the robotic hand mounted on the conventional robot arm, the robot is grasping and exploring an object using vision, tactile, and proprioceptive information (c) simulated hand interacting with a cube in the ODE physics environment (d) robotic hand interacting with a cube (e) humanoid head (f) first prototype of a quasi anthropomorphic pneumatic arm. (g) CAD design of new active vision system using the same motors and cameras as the iCub will have. (h) First prototype inspired in the iCub design.