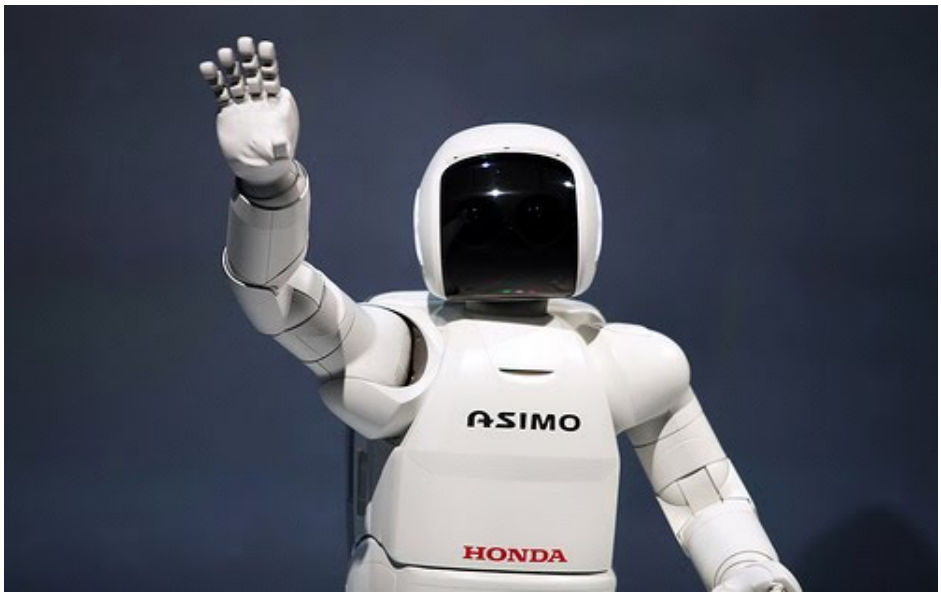
Lausanne, 06.06.2014

Semester project II: Mobile Robot modeling, Simulating and Programming.

New ASIMO



Student : Raphaël Cardinaux
Project supervisor : Michel Olivier
Professor : Francesco Mondada

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Contents

1 Introduction

This semester project consists of first modeling a mobile robot on the software *Webots* (www.cyberbotics.com), and then create an environment for this robot where we can run a short simulation to demonstrate some of its abilities. *Webots* is a software used to model, program and simulate mobile robots in a user friendly environment. The robot considered is ASIMO, the humanoid robot engineered and manufactured by Honda. This first introductory section will explain the basics about ASIMO and then talk about what informations where at my disposal to model it, and how I got them.

1.1 ASIMO

ASIMO, which stands for "Advanced Step in Innovative Mobility" is the state-of-the-art Honda humanoid robot. The first researches about walking two-legged robots at Honda R&D department started is 1986 with the E0 robot. Then came a whole set of more evolved walking robots up to ASIMO (2000) and "New ASIMO" in 2005 which is the one I have been modeling during this project. A latest version is on the way and is still in the prototyping phase which is called "All New ASIMO". Designed as a people-friendly robot with a height of 130cm and 54kg, it is intended to operate freely in the human living space and perform key tasks like walking, running, climbing stairs, carrying objects, etc. The most noticeable evolutions compared to the previous models (other than the design) are it's ability to run at a speed of 6 km/h straight, or in a circular pattern at 5km/h, and achieve tasks with human being like walking while holding hands.

1.2 Gathering informations

An important matter before and while modeling ASIMO was to gather as much useful informations as possible, mainly regarding the robot's design, its actuators and its sensors. Unfortunately Honda is not sharing many informations. Concerning the design, no URDF model was available. Therefore the pieces had to be created with a CAD system (Pro/Engineer), converted into VRML format thank to a "home made" script (*proe_vrml.c*) to adapt the content of the VRML files to what is expected by Webots (which mainly amount to erase some unnecessary lines in the files). A fairly precise design of the pieces was possible thanks to the outline drawings of ASIMO given by Jean-Luc de Krahe ("Events Coordinator& Communication" at ASIMO Studio), who was presenting ASIMO during the Inauguration of the SwissTech Convention Center at EPFL. During this event, I also had the opportunity to take close pictures of ASIMO, which was very helpful at some points during the design. Regarding the actuators stroke the main sources of information were videos were it is pretty straightforward to deduce them, and for the maximal torque, some calculations, similar robot's data and experiments made it possible to have a honest estimations. Finally, in regard to the sensors, informations were easily gathered regarding what type of sensor were on-board, however no detailed informations were to be found.

2 Modeling

This section presents all the necessary informations regarding the model of ASIMO and the rest of the world where he evolves during the demonstration.

2.1 ASIMO

2.1.1 Body parts

ASIMO's members were designed thanks the outlines drawings and pictures named before. Knowing its height and width (only data provided by Honda concerning the dimensions), the other dimensions have been calculated using a rule of three. The objects designed in ProE and imported in Webots are treated as *Shapes*, which means they only have a visual and aesthetically purpose. When imported in Webots, a body part is a multitude of shapes, which I gathered in a *Group* and placed as the children of a *Transform* (a new

coordinate system is defined for the children of the Transform node). The "method" that seemed to be the clearest to name the node in the hierarchical tree structure of the robot, was to name the *Transform* node and the *Group* in its children the same way (on the other hand, no name are given to the *Solid* nodes located at the *endPoint HingeJoint*!). In Fig. 1, find the outlines drawing which are used for the design, and in Fig. 2 all the body parts of the robot. The body part in Fig 2 have been colored and their respective names are listed in appendix A. For more details concerning the dimensions used, please refer to the VRML files.

Note :

- The body parts N, P, Q, R and W are added to add details to the model. They are built in ProE by using the parts they are blending in (for instance, the head for P, Q and W) and extruding what is unnecessary (the rest of the head for these pieces). Once imported in *Webots*, they can be put in a *Transform* node with the exact same translation and rotation coordinates as the part they are blending in, and by putting a slightly higher scale in the *Transform* node, they appear and hide what is behind. This way of proceeding is pretty efficient as it doesn't require to rebuild a new part from scratch in ProE, the position is perfect in *Webots* as some previous coordinates are reused and it is not visible from the user's point of view that actually 2 pieces are superimposed.

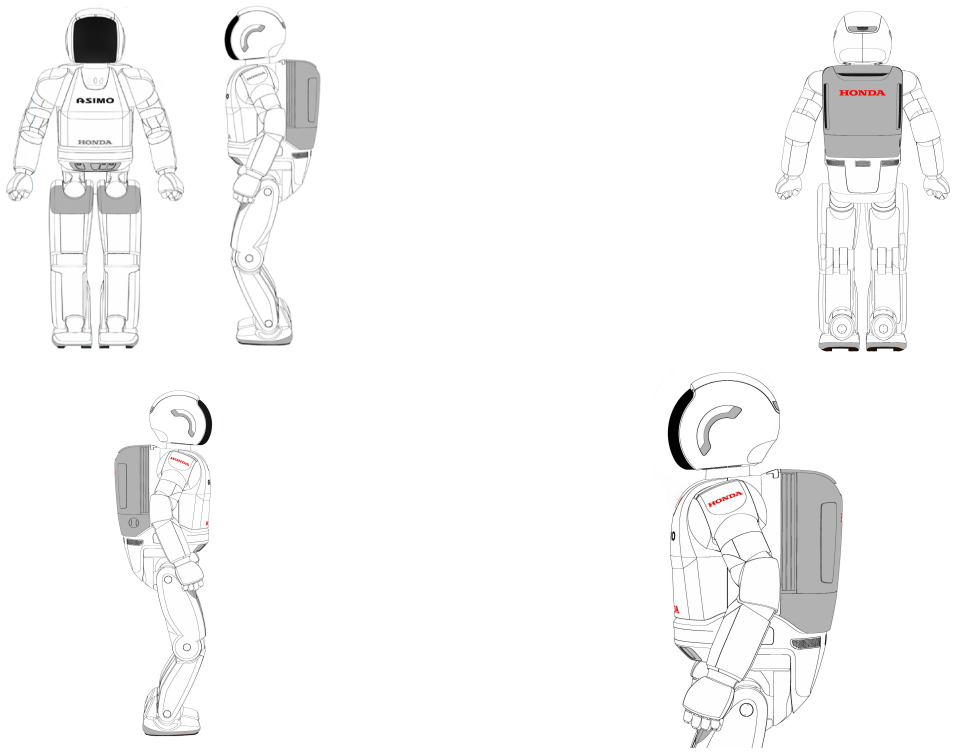


Figure 1: Some of the outlines used to compute the dimensions of ASIMO's members.

2.1.1.1 *BoundingObjects*

The *BoundingObjects* will determine the physical behaviour of our model by setting the tangible bounds of an object. Simple shapes like boxes, capsules and spheres are used to represent these bounds so that the simulation is not slown down by high computational cost (that would be encountered with IndexedFaceSet for instance). The aim is to have a model made out of *BoundingObjects* as close as possible to the real robot, only by using these basic shapes.

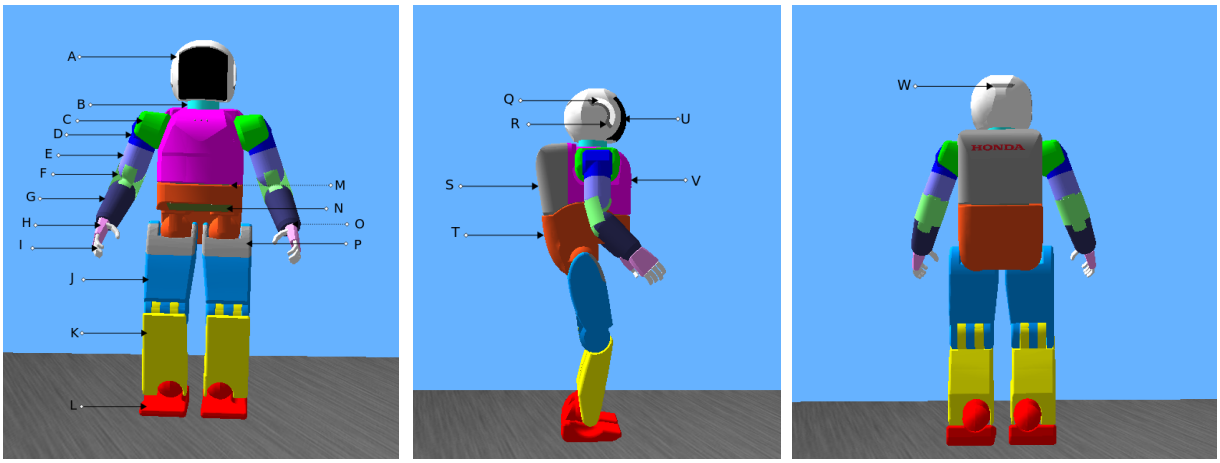


Figure 2: ASIMO's body parts and their reference letters.

Note :

- In order to keep the model as simple as possible it has been chosen not to represent with a regular *BoundingObject* the upper part of the shoulder (part C in Fig. 2) and the small intermediate part at the elbow (part F in Fig. 2). In both case, it doesn't present any particular geometrical interest. However, it has to be a *BoundingObject* so that the *physics* node work properly. Therefore, very small spheres, with almost 0 mass were added at this place as *BoundingObject*, and placed inside the *BoundingObject* which stand above or below. The same problem is encountered at the hips, the neck and the wrist : as it is a 3 DOFs joints, it is made with a *Hinge2Joint* (2DOFs) followed by a *HingeJoint* (1DOF). The *endPoint Solid* of the first *Hinge2Joint* has for only child the *HingeJoint*, that's why we also have to use this trick to make the physics node work properly. This way, it doesn't influence the mass or the physical behaviour of the model, and doesn't create internal collision. These spheres are represented in green in Fig. 3, while the regular *BoundingObjects* are in red. The names associated to the regular *BoundingObjects* can be found in appendix B.

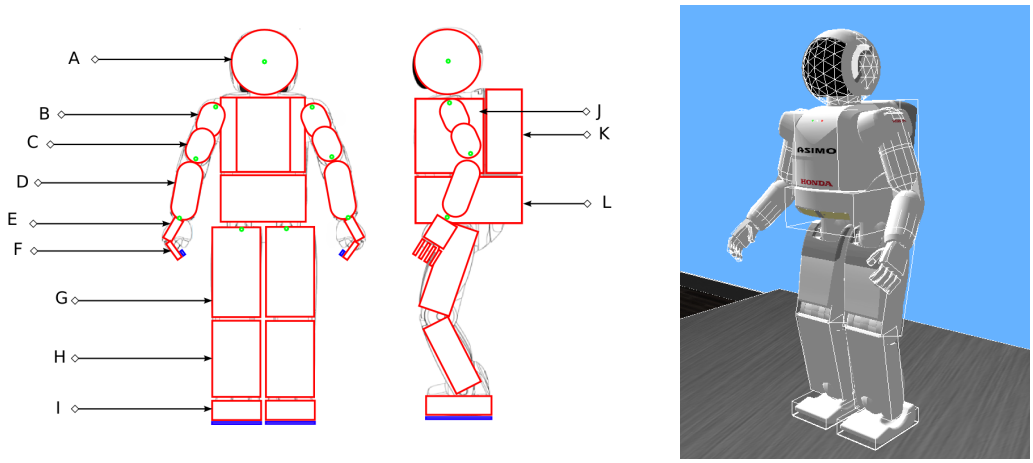


Figure 3: Sketch of the *BoundingObjects* and image of ASIMO in *Webots* with visible *BoundingObjects*. In red the regular *BoundingObjects*, in green the spheres used to make the physics node work properly and in blue the *TouchSensors* (c.f. section 2.1.3).

- Representing the fingers with only one small box does not approximate accurately the real shape of the finger. To increase the similarity, the *BoundingObject* used as touch sensors (c.f. section 2.1.3) are placed at the fingertips with a significant thickness : this makes the *BoundingObjects* a better approximation of the real finger shape, and will improve the efficiency when grasping objects (see Fig. 4). These *BoundingObjects* are represented in blue in Fig. 3.

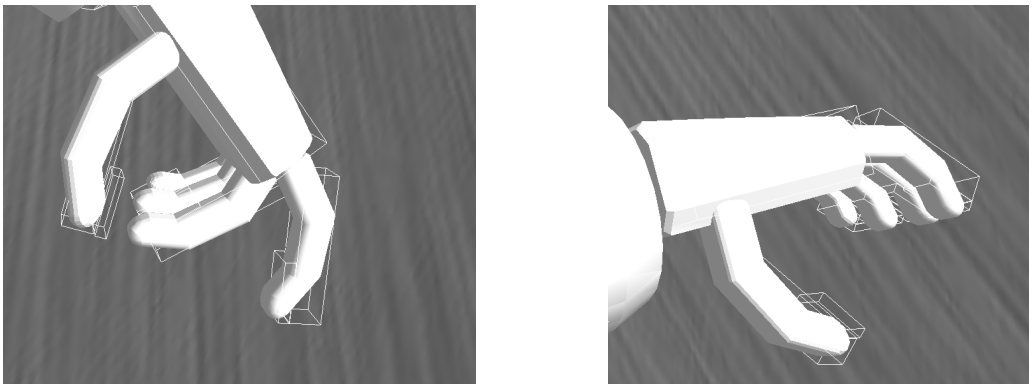


Figure 4: Fingers *BoundingObjects*.

To these *BoundingObjects* are associated a mass or a density. It is chosen to work with mass on this model, so that the problem of body parts geometrical bounds and their mass can be treated separately. In order to obtain a valuable estimation of each body part, the data of human body parts weights are considered, combined with the body parts weights of the humanoid robot Reem-c which are known thanks to the investigation of Florent Zufferey on his respective project. The result are shown in appendix B, and seems to be rather close to reality when looking at the robot behaviour in the demonstration simulation.

2.1.2 Actuators

ASIMO has 34 DOFs which are represented in Fig. 5. The actuators names corresponding to the letters in the figure are listed in the appendix C, where one can also find the min. and max. position of the actuators and their maximal torque. They are named after the name of the *HingeJoint* they belong, adding the letter *a* or *b* in case it is a *Hinge2Joint* (2 DOFs and therefore 2 actuators).

They are a few noticeable things concerning the joints and actuators which have to be raised.

- First of all, unlike its predecessors, the rotation at ASIMO's the elbows is done around two parallel axes, and this is why it is modeled as two actuators in *Webots*. However as the technical data specifies that there is still only one DOF at the elbow, both axes are activate by one actuator. This way of proceeding gives ASIMO's elbows a larger stroke than when working with only one axis. This being said, one shouldn't program these two actuators independently in *Webots* as there actually is only one actuator!
- Secondly, ASIMO's hands have only two DOFs. Indeed, the four fingers (index, middle, ring and pinky) are activated by one actuator which pull a cable which close the fingers starting with the pinky, and gives the illusion fingers can move independently. For this reason, each finger is modeled with one actuator, but one should keep in mind that they can't be commanded independently (at the moment, this feature is only possible on the prototype "All New ASIMO").

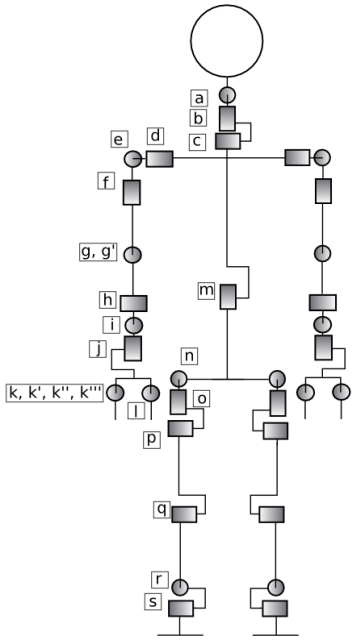


Figure 5: Graphical representation of the actuators and their reference letters (actuators orientation correspond to the model).

Figure 6: Close look on ASIMO's elbow (picture taken at the Inauguration of the SwissTech).

For those two reasons, ASIMO has 42 actuators in the model (34DOF + 2 $\times$ 3 additional actuators in the hand + 2 additional actuators in he elbows).

Concerning the strokes of the actuators, it is pretty straightforward to set them by watching videos and pictures. When it comes to make ASIMO walk, climb or go down stairs and turn around, one can see that the bounds were wisely chosen as they allow to make the model do any moves the real ASIMO can, without letting place for any incoherent moves (ex : knees bending backwards). Finally setting the maximum torque was an iterative process as, once again, no information were available. At first an estimation for the arms torque was made, supposing it should be able to accelerate a work load of 2kg at $\frac{2\pi rad}{s}$ in its hands. Then

this values were compared with the ones of Reem-c which were pretty different. The maximum Torque were adapted if the value of Reem-c were higher. For the legs, (ankles, knees and hips) the value given in the Reem-c's URDF model were way too low to even make it possible to walk properly. So at last, the torques were computed experimentally, increasing it until the motor response is fast enough to perform the tasks presented in the demonstration video. In the end a security margin is added to each torque so that the robot can also perform more demanding tasks like running. The results are shown in appendix C. In Fig. 7, one can see the process of dimensioning the actuators torque, when using a maximal torque of 50Nm, 100Nm and 200Nm for the motor *"hip_pt2_right"*. During the demonstration video : at 50Nm and 100Nm, the robot systematically falls when simply walking as the motor can't track the reference fast enough (in red the reference, in black the actual motor's position), that's why it is set to 200Nm, including a security margin (other more demanding situations have also been tested). The other legs and arms actuators are also tested and if necessary dimensioned this way.

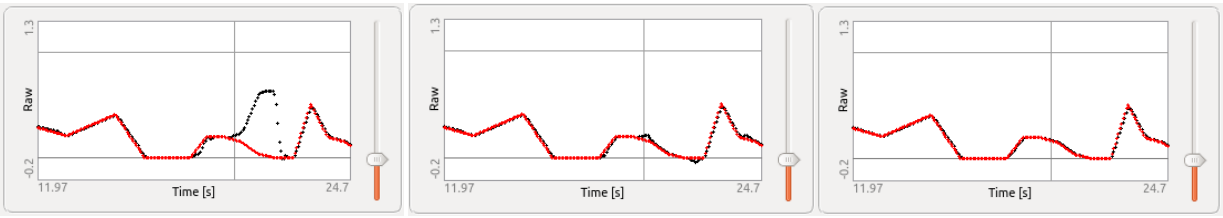


Figure 7: *Hip_pt2_right* with maximal torque of 50Nm, 100Nm and 200Nm at 24.73 s. in the demonstration simulation (in red the reference, in black the actual rotor actuator's position).

2.1.3 Sensors

The embed sensors on ASIMO are pretty common on humanoid robots and more generally on mobile robots. However no informations were available on the ASIMO's sensors data regarding for instance the resolution or the noise model, this is why there were let to their default values in *Webots*. This leads to a pretty approximative performance of the model's sensors compared to the real ones, but still gives the opportunity to the user to have substantial values to work with. Here is the list of the embed sensors :

- **Accelerometer, Gyroscope and inertial unit**
  This basic setup allow ASIMO to have the knowledge of its orientation and movement in space. It is not 100% sure whether ASIMO really has an inertial unit, nevertheless as it is a cheap device and it is very likely that is embed, it has been decided to add it. These three elements are located in the body part "BODY_DOWN", right next to the center of mass.

- **Distance sensors**
  ASIMO posses different sensors to identify its surroundings. It has 5 ultrasonic sensors in the back and 1 in front horizontally oriented with a range of 3 meters. These sensors are mainly useful to locate transparent objects, and objects that could reflect a laser. It also has one laser in front, oriented to the ground in order to detect obstacles and ground's surface, also with a range of 3 meters. An infra-red sensor with 6 rays oriented to ground is also intended to detect obstacles. These sensors are illustrated in Fig. 8 and 9. All Distance sensors have been calibrated such that it gives a linear respond between 0 and 3 meters, with a signal equal to 0 for distances of 3 meters or more, and 1000 when the distance from the sensor is null.

- **Camera**
  ASIMO is equipped with a HD camera (720 x 1080) in the head, associated to a range-finder which allow it to get depth information. This camera can for example be used to apply some machine learning algorithms to recognize objects or person if the user is familiar with these techniques. Note that the camera is by default not activated in the controller as it dramatically slows down the simulation.

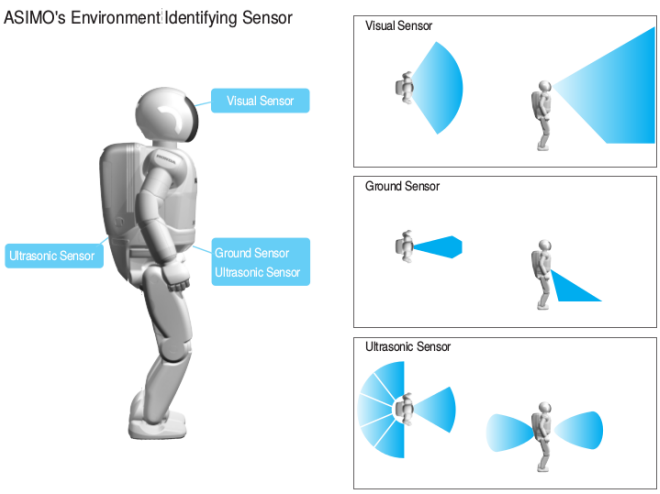


Figure 8: Asimo environment identifying sensors.

- **Force sensors**
  One axis force sensors are present in the the feet and the hands (node : *TouchSensor*). It has already been shown how the touch sensors in the hands are implemented in section 2.1.1. It basically amounts to add a *Boundingobject* with the desired orientation, knowing that the force will be detected along the z-axis.

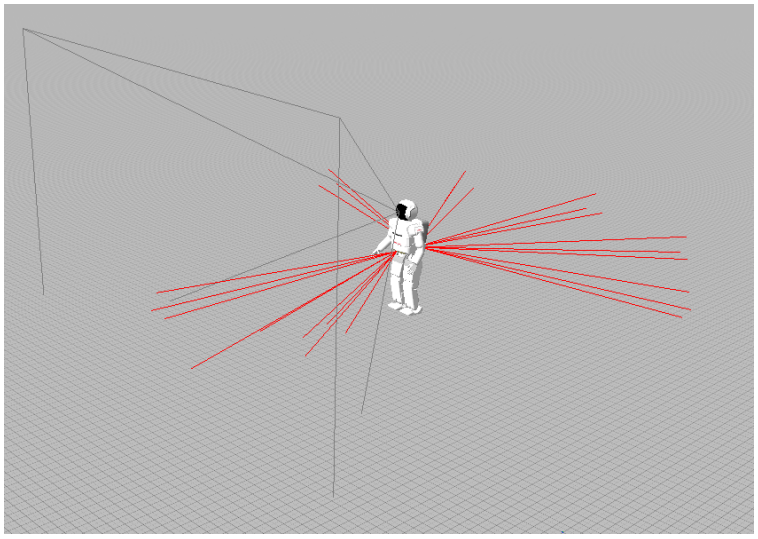


Figure 9: Actual image of ASIMO's sensors rays in red and its camera frustum in grey.

2.2 ASIMO's world

When people have the chance to see ASIMO for real (or in videos), it is most of the time on a stage during a presentation. Consequently, I decided to recreate a similar environment so that the user or spectator could more easily associate the small demonstration to something known. The world is made out of 20 chairs, a stage where ASIMO is performing, on the stage lies a small step, a ball and a goal. Behind the stage stand a huge poster of ASIMO. Some textures are used to makes the environment look more pleasant : 2 different kind of parquetry textures are used for the ground and the step and one metal texture is used for the stage.

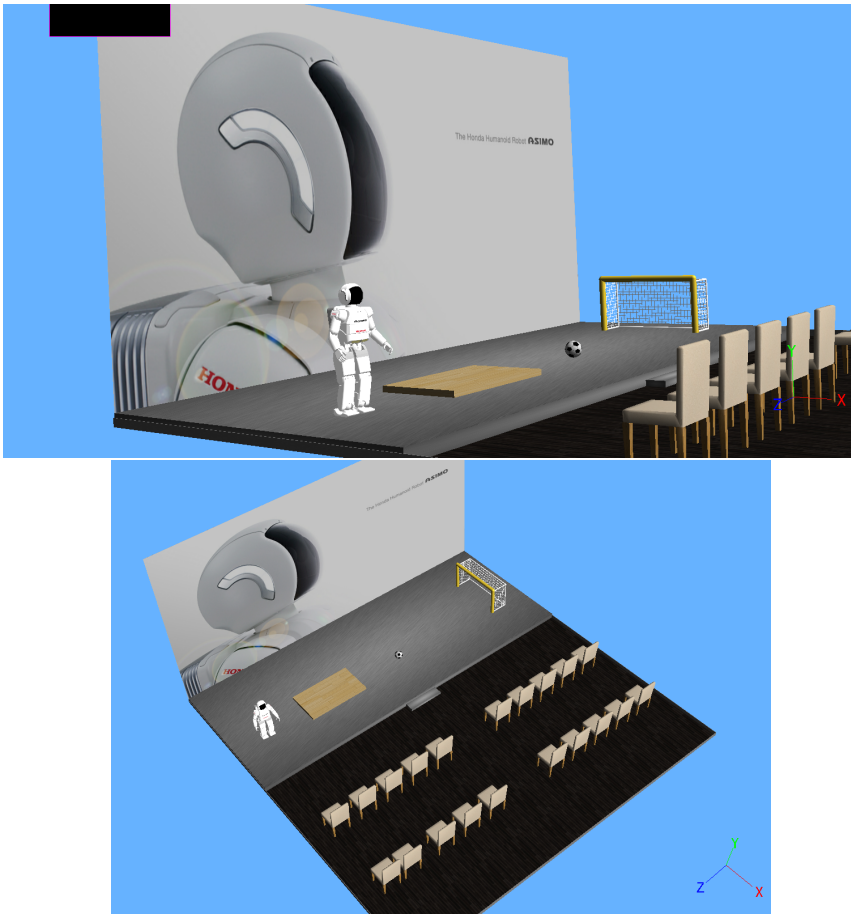


Figure 10: Screenshots of the world with 2 different orientations.

3 Simulation and Results

A small simulation presents ASIMO doing basic tasks like walking, climb up and down a step, waving at the user and scoring while standing in balance on one foot (can be seen by running the simulation in webots with the controller "asimo_controller.c". The idea is to show some of the usual moves and "tricks" ASIMO performs when doing a presentation in front of a audience. In Fig. 11, one can see the final ASIMO's design in *Webots*.

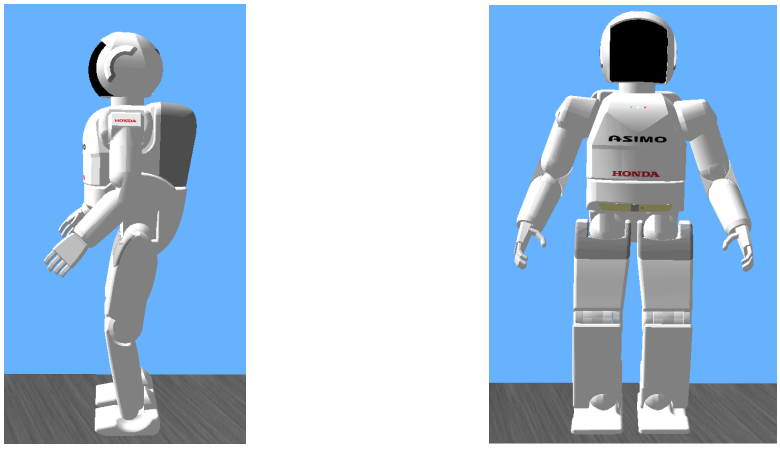


Figure 11: Results of ASIMO's design.

3.1 *Motion editor*

The *Motion editor* is a powerful tool of *Webots* which allow the user to edit motion by setting the actuators positions at different steps in time. This way of editing movements does not need to access the controller or to do any programming. As working on walking algorithm is out of the scope of this project and is anyway a project topic in itself, the *Motion editor* appeared as the ideal tool to prepare a small demonstration. Many different sequences (18 in total) are used to complete the presentation and are reusable for other purpose if one is willing to do another demo (find an exhaustive list of the motion and a short description for each of them in the appendix D).

The strategy used to make the robot walk is to always keep the projection of its center of mass on the ground inside its support polygon. This way the robot always achieve stability and do not fall on the ground. This simple strategy appears effective, even if pretty difficult to apply when climbing up or down the step. Another challenge is to avoid slipping when switching foot. Indeed, when the two feet are on the ground, transferring the projection of the center of mass from one foot to the next is hard without changing the position of the feet on the floor and usually require many tries before obtaining a satisfying result. Even if this method is purely experimental, and very time consuming, it gives convincing result in the end, and allow the spectator to get a good idea of what the robot can do. Moreover, spending time on developing movements this way is a good chance to see small details to correct that would have been left unseen otherwise, like : internal collision in movement, wrong mass repartition in the body, wrong actuator's stroke, etc.

3.2 Controller

After adding all the devices and enabling them, the controller *asimo_ controller.c* is a simple program which has two main functionalities : the first one is to run the demo by sequentially playing the appropriate motions, and then to make the robot move as the user steers it. Let first have a look at the functions the controller is made out of and its function block diagram in Fig. 13 :

- `find_and_enable_devices()`: Look for all the devices (actuators,captors, leds, etc) and enable them.
- `load_motion_files()`: Load the motion files edited in *Motion editor*.
- `start_motion(WbMotionRef motion)`: start the *motion* and wait until it ended to go out of the the function.
- `set_all_leds_color()`: Set the three leds to their respective colors (green, white and red).

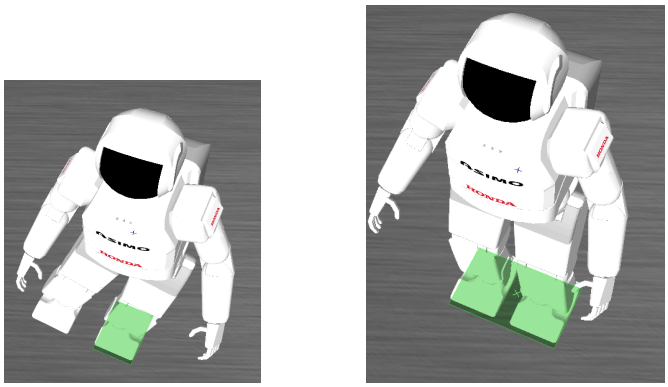


Figure 12: Center of mass (blue cross) and its projection on the ground (green cross) when standing on two and one foot. The support polygon is the green area.

- `print_help()`: Print a small help message for the user.
- `run_command(int key)`: Depending on the key which is typed and the state in which the robot find himself, this function will make the robot do one move or an other.
- `simulation_step()`: run the simulation if no special command is called and stop the simulation if a problem occur meanwhile.
- `demo()`: Run the demo by playing the sequence of motions it contains.

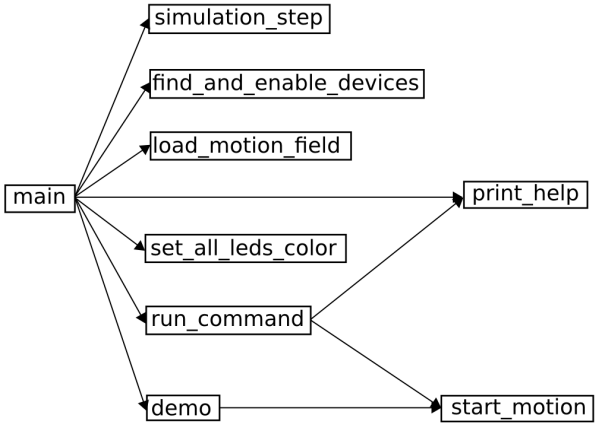


Figure 13: Function block diagram of the controller.

When the simulation is over the user can steer the robot. He can choose to go forwards, left, right or stop. To manage the changes of direction, the global variable *state* is used, which contains the state in which the robot is. *State* can only contain 3 values : 0 when it is stopped, 1 when the right foot is first and 2 when the left foot is first. Knowing this and the key pressed, two "switch" are used in the function `run_command` to chose which motion should be played, as shown in the Fig. 14.

Remark : the moves *sharp_ turn_ right* and *sharp_ turn_ left* start from a standing position with both feet parallel, that's why it goes first to state 0 before going on with the move and ending up in the same sate when the right or left key is pressed.

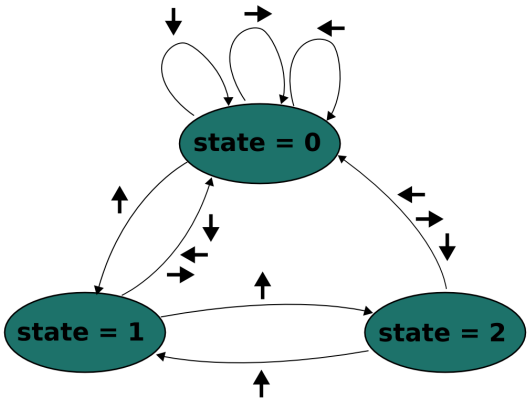


Figure 14: State diagram when the ASIMO is controlled by the user.

4 Conclusion

As the amount of data available to model ASIMO was very small, this project was mostly about observation and then experimenting to check if what was implemented is acceptable. First for the design of the body parts in ProE whom dimensions have been mostly deduced from sketches and pictures. Secondly to place this body parts in the model, with the right axis orientation and locations for the joint. Then to approximate the actuators torque with an iterative approach, and finally, to make the robot move in the same manner the real ASIMO does. The fact that it was so hard to gather informations for this robot can be partly explained by the fact that they are only a few ASIMO around the world making presentations, and these robots are not built to be sold, but mostly on a marketing purpose for Honda.

Working with *Webots* was nice, as it is very intuitive, particularly thanks to its graphical user interface which allow to directly distinguish the important informations regarding : the world, the simulation and the controller. The documentation which comes with the software (*Webots User Guide* and *Webots Reference Manual*) is very helpful to get a good understanding of the main program's features, especially thanks to well detailed tutorials. The hierarchical tree structure with nodes used to build the robot makes the robot modeling visual and therefore easier, even for complex robot like ASIMO. Regarding the controller, the fact that the user can choose among a wide range of programming languages is very convenient. It is also very appreciable that the code used on other robots is accessible, which can be used as a functional reference when it comes to implement our own controller.

To conclude, this project was very instructive as it taught me how to use a new program. It is also rewarding because the progress were visible all along the project, with the model slowly getting closer to the real robot, with finally a functional model.

I would like to seize this opportunity to thank Mr. Olivier Michel who was always available to answer my questions, and followed us during the whole process of modeling, programming and simulating ASIMO.

References

Honda (2007). *Technical information.*

Appendices

A Body parts names

In the following table are listed the names of the body parts of ASIMO, which correspond to the names of the *Transform* node and *Group* node that contain the shapes. The letters correspond to the Fig 2.
Remark : the symmetric parts always have the same name, only the ending varies from "_RIGHT" to "_LEFT".

Table 1: Body parts and their respective names

Letters	Names
A	HEAD
B	NECK
C	ARM_PT1_RIGHT
D	ARM_PT2_RIGHT
E	ARM_PT3_RIGHT
F	MIDDLE_ARM_RIGHT
G	FOREARM_RIGHT
H	HAND
I	THUMB_RIGHT, INDEX_RIGHT, MIDDLE_RIGHT, RING_RIGHT, PINKY_RIGHT
J	THIGH_RIGHT
K	LEG_RIGHT
L	FOOT_RIGHT
M	BODY_INSIDE
N	BODY_DOWN_SENSORS
O	HAND_LINK_LEFT
P	THIGH_GREY_PART_RIGHT
Q	SIDE_LIGHTS
R	HEAD_SIDE_GREY_PART
W	BATTERY
T	BODY_DOWN
U	FRONT_GLASS
V	BODY_UP
W	HEAD_BACK_GREY_PART

B BoundingObjects names and weights

The BoundingObjects are named after the names of the body part they represent in Fig. 3. Note that the weights for symmetric members is the cumulated weight for the two body parts in Table 2. The BoundingObjects which only purpose is to make the physics node work have no special name (spheres).

Table 2: BoundinObjects names and estimated weights in % of the total weight and kg

Letter	Name	Estimated weight [%]	Estimated weight [kg]
A	HEAD	4.1	2.214
B	ARM_PT2_RIGHT, ARM_PT2_LEFT	2.6	1.404
C	ARM_PT3_RIGHT, ARM_PT3_LEFT	2.6	1.404
D	FOREARM_RIGHT, FOREARM_LEFT	2.9	1.566
E	HAND_RIGHT, HAND_LEFT	1.1	0.594
F	THUMB_RIGHT, INDEX_RIGHT, ...	~ 0	~ 0
G	THIGH_RIGHT, THIGH_LEFT	15.5	8.37
H	LEG_RIGHT, LEG_LEFT	9.1	4.914
I	FOOT_RIGHT, FOOT_LEFT	2.1	1.134
J	BODY_UP	15	8.1
k	BATTERY	15	8.1
J	BODY_DOWN	30	16.2
		100%	**54 kg**

C Actuators, stroke and maximum torque

Table 3: HingeJoints with their minimum and maximum positions and maximum torque

Ref.	Joints names	Max. Torque [Nm]	minPos. [rad]	maxPos. [rad]	Stroke [rad]
a	neck_pt1_a	3.5	-0.3	0.3	0.6
b	neck_pt1_b	3.5	-1.57	1.57	3.14
c	neck_pt2	3.5	-0.2	0.4	0.6
d	shoulder_pt1_right, shoulder_pt1_left	45	-1.57	3.14	4.71
e	shoulder_pt2_right, shoulder_pt2_left	30	-0.2	1.1	1.3
f	shoulder_pt3_right, shoulder_pt3_left	15	-1.57	1.57	3.14
g	elbow_pt1_right, elbow_pt1_left	15	-0.2	0.8	1
g'	elbow_pt2_right, elbow_pt2_left	15	-0.4	0.8	1.2
h	wrist_pt1_right_a, wrist_pt1_left_a	3	-0.3	0.3	0.6
i	wrist_pt1_right_b, wrist_pt1_left_b	3	-0.4	0.4	0.8
j	wrist_pt2_right, wrist_pt2_left	3	-1.57	1.57	3.14
k	index_joint_right, index_joint_left	2	0	1.57	1.57
k'	middle_joint_right, middle_joint_left	2	0	1.57	1.57
k"	ring_joint_right, ring_joint_left	2	0	1.57	1.57
k"'	pinky_joint_right, pinky_joint_left	2	0	1.57	1.57
l	thumb_joint_right, thumb_joint_left	2	-0.4	0.5	0.9
m	body_joint	45	-0.3	0.3	0.6
n	hip_pt1_right_a, hip_pt1_left_a	200	-0.4	0.3	0.7
o	hip_pt1_right_b, hip_pt1_left_b	200	-1.57	0.5	2.07
p	hip_pt2_right, hip_pt2_left	200	-0.2	1.3	1.5
q	knee_right, knee_left	150	-1.6	0.4	2
r	ankle_right_a, ankle_left_a	150	-0.3	0.3	0.6
s	ankle_right_b, ankle_left_b	150	-0.8	0.6	1.4

D Motions

List of the motions used in the demonstration and a short description :

- **walking_start** : start walking with the right foot first.
- **walking_leftgofirst** : at the beginning the right foot is first, the left foot is back in a walking position. Then the robot put its left foot in front.
- **walking_rightgofirst** : same as "walking_leftgofirst" with inverted foot.
- **walking_stoponright** : the right foot is in front in a walking position. The left foot is brought in front next to the right foot in a standing position.
- **walking_stoponleft** : same as "walking_stoponright" with inverted feet.
- **wait** : does not do anything for two seconds in a standing position.
- **wave** : wave with the arm.
- **wave_end** : goes from a waving position to a regular standing position.
- **wave_start** : goes from a regular standing position to a waving position.
- **sharp_turn_right** : do a 90 deg turn the right.
- **sharp_turn_left** : do a 90 deg turn on the left.
- **stairs_up_start** : climb on step up.
- **stairs_down_start** : climb on step down.
- **score** : go on balance on one foot and score.
- **victory_dance** : perform a victory dance!
- **victory_dance_start** : go from a regular standing posture to a dancing posture.
- **victory_dance_end** : go from a dancing posture to a regular standing posture. bend;