Construction of the Humanoid Robot InMoov

George Hartt[1,*]

[1] *Intern at CTU in Prague, Faculty of Mechanical Engineering, Technická 4, 166 07 Prague 6, Czech Republic*

Abstract

The article introduces the humanoid robot InMoov. Experiences from its assembly, advantages, and disadvantages of the construction are described. This is a 3D-printed robot with approximately 45 joints. The robot is of human size. The robot described was assembled in the laboratories of the Institute of Instrumentation and Control Engineering.

Additionally, attempts at implementing controls in a rigid 3D body simulator using ROS are also documented below.

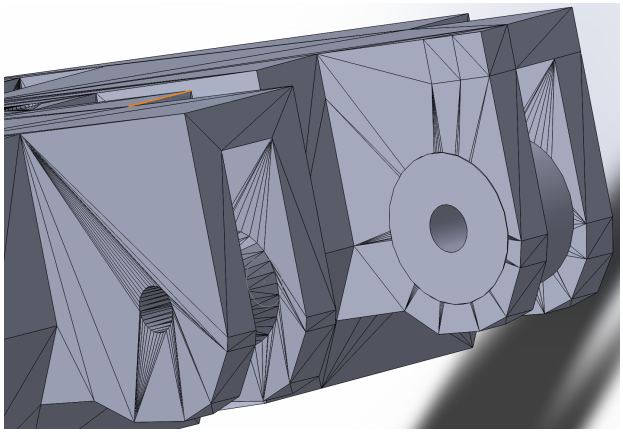
Keywords: InMoov; robotics; 3D-printing; controls; ROS

1. General Overview

InMoov is an open-source humanoid robot made by Gael Langevin, a French sculptor and designer. The main idea behind InMoov is making it open source. This was achieved by designing the robot to be predominantly built from 3d printed components from a printer with an area of only 12x12x12 centimetres.

The robot is actuated by servos, specifically HK15298B, Hitec HS805BB and MG996. The low-level control of these servo motors, among other peripherals, is implemented using a pair of Arduino Megas.

The program controlling the robot is implemented in ROS[1], an open source robotics middleware environment. This allows for very powerful motion planning among other features.



Pic. 1. *on the left: before introducing geometry; on the right: after introducing geometry.*

2. Software

2.1 Control

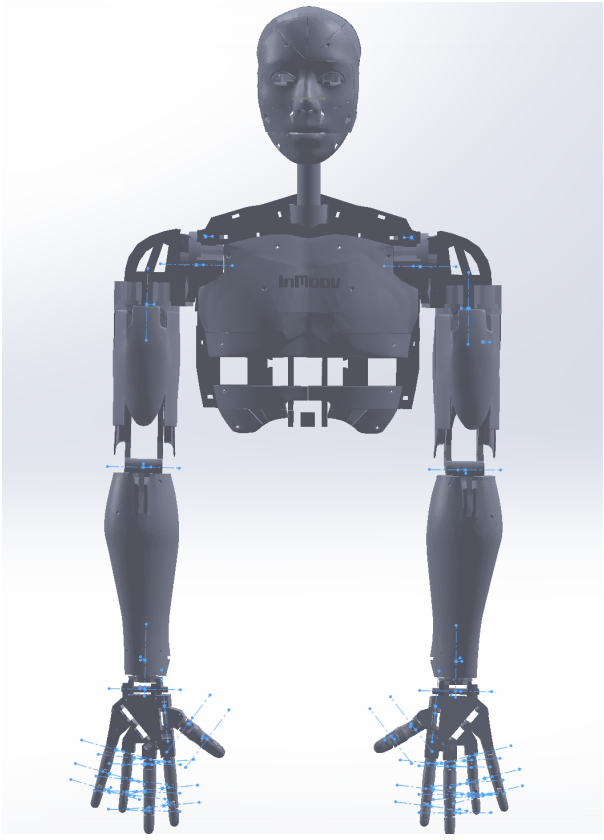
2.1.1. Overview

First, a Solidworks model of InMoov was assembled from STL files available online. Following that, a URDF file was generated from this model using the plugin *sw_urdf_exporter*[2], which enabled the robot to be visualised in *rviz*. *MoveIt!* was then used to enable motion planning. Finally, the joint angles were scraped and converted to degrees, which are relayed to the micro controller.

2.1.2. Solidworks model to URDF

To create the Solidworks model, STL files were collated into individual static parts, such as *r_bicep*, *r_wrist*, *r_hand*, and so on. During this step, as many redundant parts were removed to reduce the number of polygons in the model. For this purpose, the neck mechanism was also replaced with a simple alternative to further reduce the triangle count.

Over 50 of these STL parts were then altered to introduce real geometry, such as cylinders and holes, as can be seen in Picture 1. These were then exported as Solidworks parts and assembled into a Solidworks assembly to create a bare-bones model of the InMoov robot (see Picture 2).



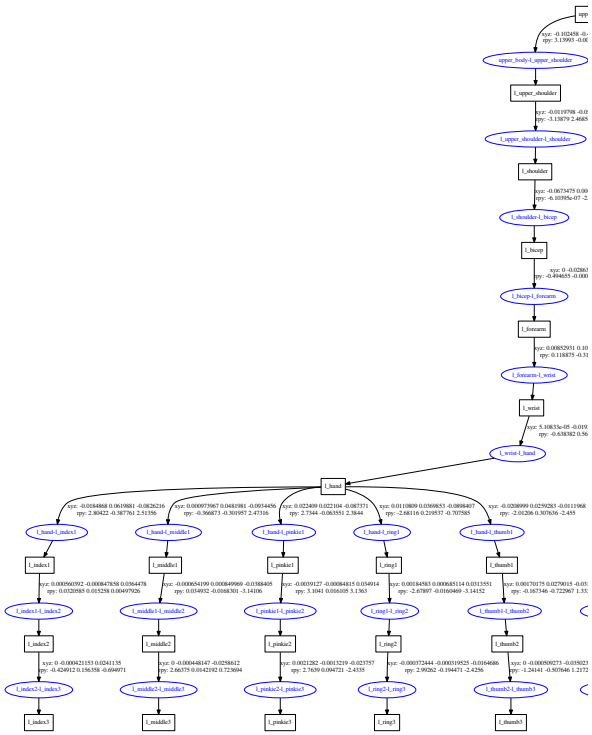
Pic. 2 *Solidworks model of InMoov with URDF references (blue axes) generated after export.*

The plugin *sw_urdf_exporter* was then used to generate a URDF file. This involved adding all the parts to a tree structure, with the root node being the upper-body, then progressively adding the parts up to the finger tips. The general outline of this tree structure can be seen in Picture 3. Joints or links were then configured, namely their type, upper and lower limits, orientation, origin as well as name, required moment and speed of rotation. The file was then exported and saved to disk, at which point it could be loaded into *rviz* and *gazebo*.

2.1.3. 3d Rigid Body Simulation

A SRDF file was then generated using *MoveIt!* and the simulation was set up. This involved creating 4 groups—*r_arm*, *l_arm*, *r_hand*, and *l_hand*— as well as specifying the kinematics algorithm, for which *KDLKinematics* was chosen.

A few pre-defined poses, such as moving the right arm up and down, were then added. All the inverse-kinematics, redundancy handling and collision detection are handled automatically after the initial setup. At this point, the simulation had fully integrated motion planning.



Pic. 3. *left-hand side of the tree structure generated from the URDF file, from the body to the fingers (segmented into 3 parts). To view in full, see http://docdro.id/sFwwkwg.*

2.2. Firmware

The firmware was written using the Arduino IDE. Its task is to interpret messages from the computer running ROS and carry them out on the hardware.

2.2.1. Communication

For the low-level implementation, USB serial lines were used to enable communication between the Arduinos and the desktop computer. The decision to use USB serial over other protocols, such as SPI, is largely due to it's relative simplicity, as well as communication speed requirements not being large enough to justify more complex implementations.

For the communication between the firmware and simulated robot, *joint_state_publisher* and *rosserial-arduino* were used to reflect the simulated robot onto the physical one. This was done by subscribing to the topic *joint_state_publisher* and getting the joint angles.

At the moment the communication is one-way only due to the robot's open-loop implementation.

2.2.2. Protocols

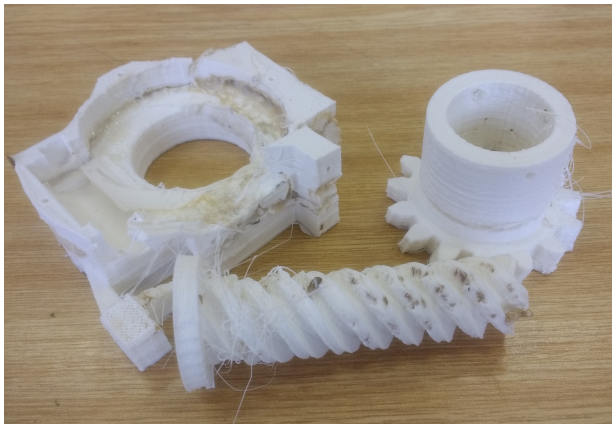
Communication between ROS and the Arduino firmware was implemented using a strict g-code-like packet format: [1-ascii-character][2-digit-unsigned-int][3-digit-unsinged-int]. For example, to fully extend the right bicep, the packet 'S08000' would be sent, where the first character signifies Servo, 08 corresponds to the servo controlling the bicep mechanism, and 000 represents the angle of 0 degrees.

The software end-stops are enforced in the firmware to prevent the robot from breaking in case of a bit-flip occurring on the USB bus, which is another reason for using a strict packet format.

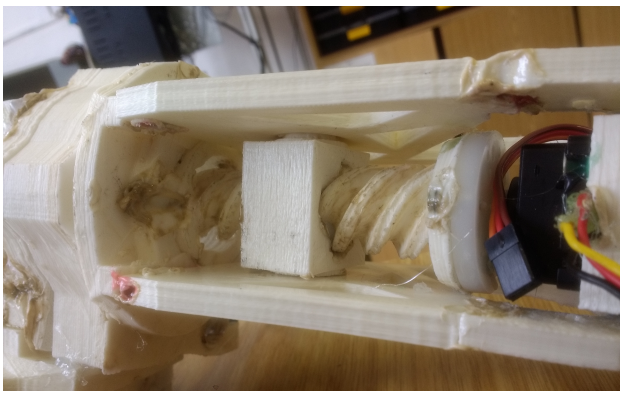
3. Hardware

3.1 3D Printed Parts and construction

InMoov is constructed mainly from 3d printed parts, with the aim of staying open-source and being accessible to anyone. This eliminates the need to search for unconventional parts that may not be available everywhere. However, in practice this introduces the issue of parts being almost-unusable for mechanical constructs, such as gear boxes and worm-gears, if the printer is not properly calibrated. This can be seen below in Picture 4.



Pic. 4. *a few parts fresh off the printer.*



Pic. 5. *a worm gearing assembly after repairs.*

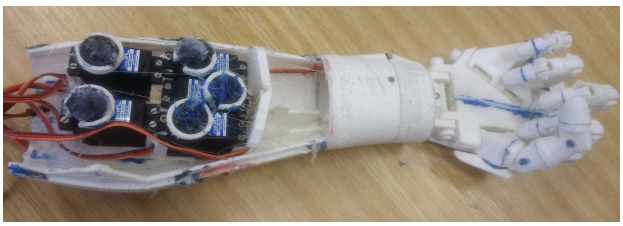
To make the parts usable, laborious re-sculpturing had to be done using a soldering iron, files, and a hot-glue gun. This had mostly positive results, as can be seen above in Picture 5.

3.2 Actuation

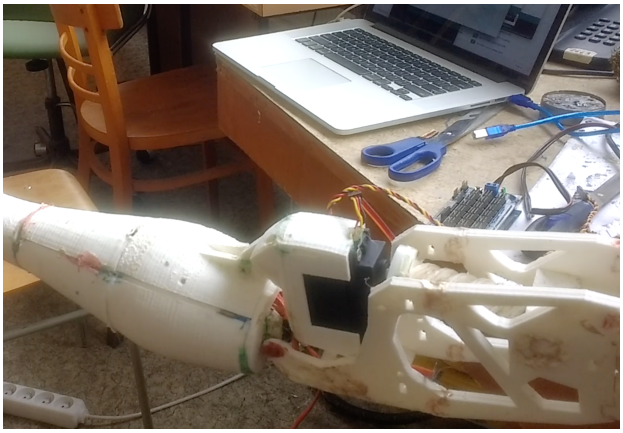
The robot InMoov has over 45 joins, with 17 of them being in each hand and 4 being located in each arm. To actuate the movement in each one, different mechanical systems were employed.

3.2.1. Fingers

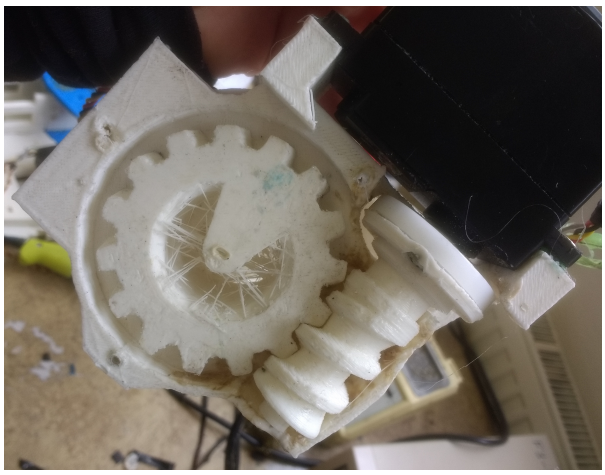
The fingers on each hand are actuated using a 5521MG servos in conjunction with a 200lb non-stretchable fishing line. This line is tensioned and attached to the servo's output face, as can be seen in Picture 6. This has the effect of either contracting the finger at the maximum position, or relaxing it in the minimum position.



Pic. 6. *actuation of the fingers, currently in max. position.*



Pic. 7. *actuation of bicep; relaxed*



Pic. 9. *worm drive actuated using HS805BB servo*

3.2.2. Bicep

The forearm moves using a worm gear assembly. It is actuated using a HS805BB servo and allows the forearm to rotate from 0 to 70 degrees about its joint with the bicep. It uses an external potentiometer to ensure the servo can rotate continuously until it reaches the angle specified at the joint. (See Pictures 5, 7, and 8.)

3.2.3. Shoulder and Forearm

The other type of mechanism used to actuate movement is a worm drive transmission, as can be seen in Picture 9. This mechanism is used to rotate the forearm and shoulder, as can be seen in Picture 2.

3.2.4. Continuous Rotation with Servos

In order to make the joints extend to their full capability, the servos had to be modified to receive potentiometer input from the joint. To do this, the servo was first disassembled, and the main gear's mechanical end-stop was removed. Following that, the servo's circuit board was taken out, and the potentiometer connected to the shaft was extracted. The connecting wire was then lengthened, and the potentiometer was taken out of the servo so that it could be used externally.

Pic. 8. *actuation of bicep; contracted*

4. Plans for the Future

4.1 Image Recognition

InMoov is equipped with 2 2-Megapixel cameras in the eyes that are connected to the computer via USB. At the moment, they are not used for any purpose other than streaming live footage from the robot's perspective.

A goal for the future is implementing image recognition using frameworks such as OpenCV to do functions such as face-tracking, object tracking, and others. Because there would be 2 cameras working in conjunction, the distance between the robot's head and the object could also be triangulated. Eventually, this could be used in collaboration with the control and motion planning in ROS to achieve object manipulation when combined with a XBox Kinect.

4.2 Controlling the Robot online

As a result of InMoov being controlled through ROS, it would be possible to implement controls through an online interface. This could be done by by having an online simulation (separate from the one running in the desktop PC), then send send joint information from this online simulation to ROS which would then affect the native simulation and subsequently the real robot.

5. Bill of Materials [4]

5.1 Electrical

- 2x Arduino Mega
- 11x 5521MG Servo
- 10x HS805BB Servo
- 21x 100cm 3-braid copper wires
- 2x 2 Megapixel cameras

Table 1. *Information about the servos used.*

Servo	Stall Torque 4.8V/6V (kg.cm)		Usage
HS805BB [5]	19.73	24.70	Actuation of larger mechanisms such as transmission gears
5521MG [6]	17.25	20.32	Actuation of fingers and wrist via strings & gears, respectively

5.2 Mechanical

- 200lb non-stretchable fishing line
- 7kg PLA filament
- Assortment of M3-5 bolts & nuts
- Metal pins for finger joints

6. Conclusion

Given the materials I had to work with and being relatively new to mechanics, I believe that this project was a great success overall and an exciting learning process. In the future, I hope to see more work being done on this robot by other people to build upon the groundwork that I had implemented over the course of this internship.

For more information detailing the project, as well as all the source-code for the Arduino and ROS software, please visit https://github.com/nyxaria/inmoov.

Acknowledgement

A special thank you to doc. Ing. Martin Novák, Ph.D. for giving me the opportunity to intern at ČVUT Faculty of Mechanical Engineering - I really appreciate it!

References

[1] *ROS.* ROS [online]. [feeling. 2018-03-28]. Available from: http://www.ros.org/

[2] *sw_urdf_exporter.* SolidWorks to URDF Exporter [online]. [feeling. 2018-03-28]. Available from: http://wiki.ros.org/sw_urdf_exporter

[3] *rviz.* rviz [online]. [feeling. 2018-03-28]. Available from: http://wiki.ros.org/rviz

[4] *InMoov.* InMoov [online]. [feeling 2018-03-28] Available from: http://inmoov.fr/default-hardware-map/

[5] *Servo City.* Servo City [online]. [feeling. 2018-03-28]. Available from: https://www.servocity.com/hs-805bb-servo

[6] *JX Servo*. JX Servo [online]. [feeling. 2018-03-28]. Available from: http://www.jx-servo.com/English/Product/3015242947.html