First Project: Kinematic and Static Analysis of a Humanoid Robot, NAO H25

Dr. Mehdi Tale Masouleh, Payam Ghassemi
Deadline: 1392/02/01

Advanced Robotics - Semester II (2012/2013)
Human-Interaction Laboratory (TAAR-Lab)
University of Tehran
If you want to live your life in a creative way, as an artist, you have to not look back too much. You have to be willing to take whatever you’ve done and whoever you were and throw them away.

- Steve Jobs

Introduction

First and foremost, the study of any aspects of robotic mechanical systems requires a comprehensive investigation on their kinematics and statics modeling where the former properties are related by duality concepts. Through this project, it is expected that the kineostatics modeling of NAO H25 will be investigated upon following the instructions provided in what follows. NAO is a humanoid robot developed by Aldebaran Robotics and RoboCup competition included NAO Standard Platform League since 2008. Aldebaran Robotics has provided various version of robots in various categories. The most advanced version of this robot is NAO H25 v4 which has some significant improvement rather than previous models. NAO H25, Fig. 1, is known as academic version and has 25 Degree-of-Freedoms.

The installed OS on NAO is OpenNAO. OpenNAO is an Embedded GNU/Linux distribution based on Gentoo which is specifically developed for NAO. It is possible to have a remote access to this OS through ssh or FTP, which interacts with embedded software. Aldebaran Robotics provides some desktop software packages which ease the programming or accessing to several resources of NAO, e.g., Choregraphe and Monitor.

Users can develop new modules and programs using SDKs which are provided by Aldebaran Robotics. The programs can be executed directly on NAO, embedded version or executed remotely on remote system and send the commands to and receive data from NAO. Various programming languages are supported; C++, Python, .NET, Java and Matlab. Whereas, only C++ and Python languages are capable to produce the embedded versions. Also, there are APIs for these two languages which are provided by Aldebaran Robotics.

Prerequisites

This project requires to install the following software:

1. CMake 2.8.3+
2. qibuild
3. Python 2.7
4. C++ SDK (naoqi-sdk-1.12.5-win32-vs2010)
5. Microsoft Visual Studio 2010 (or 2008)
6. Choregraphe (optional)

Reading Materials

NAO H25

The main source to gain some insight about NAO H25 is the NAO Software Documentation (the latest version 1.14.1).
Start-up

If you want to know how to connect NAO and configure it, you should read the section 'NAO/Configuring NAO'.

Hardware Specifications

All hardware specifications of NAO v4 such as kinematics, sensors, motors and etc are accessible through the section ”NAO/NAO Hardware”. Also, you can get a deeper understanding about the NAO by using Choregraphe and the SolidWorks model.

C/C++ Programming (General Study)

2. Practical C++ Programming by Steve Oualline, O’Reilly.

C/C++ Programming (NAO APIs)

Read the section ”NAO Software Documentation/Programming Guide/SDK/Cpp” and familiarize yourself with three NAOqi modules APIs (Core, Motion, Sensors) from here:

1. References/NAOqi Core
2. References/NAOqi Motion
3. References/NAOqi Sensors

One of the key task which is essential for you to complete this project is accessing to NAO states (sensor data); the NAO states can be accessed via the ALMemory module and the key functions of this module are getData() and getDataListName(). Also, reading the section ”References/NAOqi API/DCM/Architecture of the pref file Device.xml / of key/values in ALMemory” is essential.

qibuild

qibuild aims to make compilation of your sources easy. It manages dependencies between projects and supports cross-compilation. You will find useful information about using qibuild in the section ”NAO Software Documentation/Programming Guide/SDK/Cpp/Using qibuild with Aldebaran packages”. But for this project only the following commands are necessary to know:

1. qitoolchain create ToolChainName
2. qibuild init
3. qibuild config --wizard
4. qibuild create ProjectName
5. qibuild configure ProjectName
6. qibuild make ProjectName
7. qibuild open ProjectName

1For matters regarding access to the SolidWorks model and other materials you may contact your TA.
Figure 1: Overview of NAO H25.

Problems

1. Kinematics Analysis:

   (a) Provide the D-H parameters for NAO H25 which completely defined the pose of robot (assumption to simplify: the right foot is fully fixed on the ground). You should use the D-H convention which is introduced in [2]. Also, attach the coordinate frames on NAO model in SolidWorks.

   Hint: Humanoid can be considered as a tree-type multi-body system which included five open-chain branches. One of the appropriate ways to write the kinematics is describing each branches relative to the float-based (Torso of NAO); the problem is divided into five sub-problems, the kinematic of five open-chain serial systems.

   (b) Consider the following trajectory in joint-space for \( q_i, i = 1, 2, \ldots, 6 \):

   \[
   \begin{align*}
   q_1 &= -18 \sin \frac{\pi t}{5} \\
   q_2 &= 18 \sin \frac{\pi t}{5} \\
   q_3 &= 0 \\
   q_4 &= -0.0001 t^3 + 0.0218 t^2 - 1.453 t + 27.49 \\
   q_5 &= -0.0008 t^3 + 0.1142 t^2 - 3.778 t + 35.168 \\
   q_6 &= 0.0002 t^3 - 0.0351 t^2 + 1.396 t - 5.514
   \end{align*}
   \]

\[^2\text{You are not allowed to use other works (papers, friends, books and etc), you must provide your original solution.}\]
where $0 \leq t \leq 5$, all the dimensions are in degree and

\begin{align*}
q_1 &= LAnglePitch \\
q_2 &= LKneePitch \\
q_3 &= LHipPitch \\
q_4 &= RHipPitch \\
q_5 &= RKneePitch \\
q_6 &= RAnglePitch
\end{align*}

Now using the D-H parameters, plot the FKP solution for the two End-Effectors, namely Torso and LLeg (Fig. 2a), for the foregoing trajectory.

2. **Static Analysis:** To verify your model with the real one of NAO, consider the following example. Using "Cartesian control API" perform the following trajectory which given in SPACE_NAO³:

\begin{align*}
x &= 0 \\
y &= \quad (13) \\
z &= 300 + 20\sin\frac{\pi}{2}t \\
&= 300 + 20\sin t \\
&= 300 + 20\sin\pi t
\end{align*}

where $0 \leq t \leq 4$ and all the dimensions are in mm.

(a) Implement the trajectory on NAO and retrieve the sensor values to compare with the theoretical results which will be obtained in the upcoming sections. It should be noted that NAO can not give the torque values, so a direct comparison is not possible. Thus, instead of retrieving the

³“This is average of the two feet positions projected around a vertical z axis. This space is useful, because the x axis is always forwards, so provides a natural ego-centric reference.”
torque values, the electric current values of motors are considered. Retrieve and plot the following parameters:

i. The joint angles of the legs,
ii. The electric currents of motors of the left leg,
iii. The values of all FSRs,
iv. The acceleration values of Torso.

(b) Determine the Jacobian of the left foot (LLeg) with respect to the Torso.

(c) Applied the obtained joint values to your FKP equation and obtained $p = [x \ y \ z]^T$ of Torso with respect to the frame attached to the right foot.

(d) Plot the torque of actuators of the left leg (LAnklePitch, LKneePitch and LHipPitch) for the above trajectory. For this purpose, you should consider:

$$\tau = J^T F$$

where $\tau$ and $J$ are the joint torques and the obtained Jacobian matrix in section (b), respectively. $F$ is the applied forces to the End-Effector (the values of FSRs of the left foot).

**Academic Honesty**

You should know that your honest act and hardworking are more important than completing all items of the project. Thus, answer the problems honestly and avoid plagiarism:

"Plagiarism is defined as use of intellectual material produced by another person without acknowledging its source, for example:

1. Wholesale copying of passages from works of others into your homework, essay, term paper, or dissertation without acknowledgment.

2. Use of the views, opinions, or insights of another without acknowledgment.

3. Paraphrasing of another person’s characteristic or original phraseology, metaphor, or other literary device without acknowledgment.”

**References**


---

4Force sensitive resistors