Research Opportunities with the NAO Robot

John Dulin

February 12, 2012

0.1 Introduction

This paper is a brief overview of the NAO robot platform's achievements, capabilities, and opportunites for research, which includes the software tools available for the robot and its hardware specifications. We are exploring possible research projects for the Air Force Research Lab, which owns the NAO robot and is funding this research through the Summer at the Edge program at its Discovery Lab. Therefore, the research ultimately pursued will be chosen by the Air Force and the lab directors.

The NAO Robot is a versatile platform, both in hardware and software. The humanoid stands 58 centimeters tall (just under 2 feet), weights approximately 9.5 pounds, can walk for 90 minutes on one battery charge, has 21 to 25 degrees of freedom, and uses a x86 AMD Geode 500 Mhz CPU for computation. The NAO's grippers are hand-like in their joints, although each hand has only two fingers and one thumb. The robot's two CMOS cameras, inertial measurement unit, four ultrasonic sensors, and four microphones give it powerful abilities for sensing and interacting with the environment, including facial and shape recognition, voice recognition, sound localization, and stability and positioning within space.

0.2 Tools for the NAO platform

Furthermore, the software tools that the NAO can work with are robust. In terms of programming languages, the NAO offers developers the choice of C++, C, C#, Python, Urbi, Matlab, Java, Visual Basic, F#, and .NET, all running on the NAOqi framework, the NAO's embedded software platform. This also facilitates compatability with the Microsoft Robotics Studio, Cyberbotics Webots, Gostai Urbi Studio, ROS, and of course Aldebaran's SDK. The two platforms we will explore are the C++ and Python SDK APIs for the NAO, because of personal experience, the speed of Python prototyping for NAO behvaior, the power of C++ for building NAO services, their cross-language compatability in modules, and their use in both the NAO SDK and ROS NAO stacks. As of 2011, a significant part of the NAOqi source code has been open-source. Tools for NAO development are received with membership in the elite NAO developers' program, a 4,800 dollar investment, or by purchasing a full price NAO at approximately 15,000 dollars for institutional research.

Aldebaran also provides a graphical programming environment for the NAO. Aldebaran's Choregraphe development tool is a cross-platform application that allows users to simply and intuitively edit NAO's movements and behaviors. This program includes a GUI that can be used to teach the NAO how to recognize images, react to events or input, and move in a certain way. This way, behaviors can be built for the NAO using a flow diagram. Choregraphe has a library of built-in abilities that can be scripted to the NAO, making it easy to have NAO perform simple actions. The simple and graphical manner of building NAO behavior in Choregraphe can be compared to LabView. However, Choregraphe does not help build more complex functionality for the NAO, and is primarily a tool for rapid prototyping and testing.

The ROS stacks for the NAO are entirely coded in Python. The nao_common and nao_robot ROS stacks built by the Albert-Ludwigs-Unversitaet in Freibrug Germany include remote control and core driver packages, respectively. The nao_robot stack provides basic functionality for NAO development with ROS nodes. Nao_common raises that functionality with tools for remotely controlling the NAO with a computer. The nature of ROS means that the functionality of NAO be rapidly expanded on these nodes.

The NAO SDK based on NAOqi is cross-platform and cross-language, with an identical API for both C++ and Python. In general, programming methods are the same for each language. If properly created, Python and C++ modules (classes within libraries) can be called from anywhere. This is possible because of NAOqi's introspection, meaning it knows all available API functions. The NAOqi itself is a broker, such that it looks up modules for the NAO whenever and wherever they are called. The memory of the robot is the thread-safe ALMemory array, an array of ALValues that can be written to or read from by any module for data from sensors and joints, events, and micro-events.

0.3 Overview of Past NAO Research and Ideas for Projects

Because of the versatility of its hardware and software, the NAO can be used in a large set of research projects. Some examples include NAO applications that have the robot review flash cards with a person, play a game of Connect 4 with an opposing player, play Rock, Paper, Scissors (Lizard, Spock), mimic the movements of a human through Kinect sensor integration, use handheld tools and carry objects through Kinect teleoperation, and autonomously plug itself into a power outlet when its battery is low. A majority of these applications have been developed by individuals within the NAO Developer Program, as Aldebaran mainly just provides the robot and software platforms for independant developers to build functionality on.

Although the computational power of the NAO is limited, cloud robotics have shown great potential in improving its vision and voice recognition. An example of this is the Connect 4 game that a NAO was able to play by handling some of the computations remotely, and of research that the cloud robotics infrastructure GostaiNet is assisting on robot interactions with children. An individual developer harnassed the power of Google's Speech-to-Text application found in Android phones to have a NAO translate speech that it heard into text. Cloud robotics offer incredible potential for making robots more powerful, more energy efficient, and lighter by loading computation off to the cloud.

Some of the most important work done on the NAO has been done on the Uni of Freiburg's 'Osiris' NAO platform, a NAO built with a 2D laser range finder for localization in complex, arbitrary environments. Localization of humanoid robot platforms has always been difficult because the movement of a human greatly increases the noise and complexity in sensor data. The work done on Osiris was used to build a full navigation stack for humanoid ROS robots, including a footstep planning library included in the alufr-ros-pkg for ROS.

So robotics research using the NAO should take advantage of its capabilites, and possibly even bolster them with some modifications. Our ideas include building a NAO application that allows it to autonomously identify, obtain, and use simple tools. This would be similar to the NAO behavior shown by Taylor Veltrop, a NAO Developer, except that the NAO would be acting autonomously instead of being teleoperated by a person and a Kinect sensor. From a programming perspective, this is a very difficult and ambitious task. It would require advanced machine vision, control, and decision making from the NAO, especially when objects where positioned, moved, or obstructed in a way the NAO did not expect. It is likely this endeavor would require a massive amount of trouble-shooting, cloud computations, and optimisation, and it is possible the project could not be perfected in a summer. However, if successful, the software would be a breakthrough for the NAO platform and the research would explore solutions to one of the most important obstacles to creating useful household robots.

Other research projects could involve the robot's localization and sensing abilities. Of particular interest is the usefulness of integrating remote sensors with the NAO and robots in general. For instance, having remotely placed laser range finders, Kinects, microphones, and/or other vision sensors collect data on an environment that the NAO will explore but does not yet have sensor data on. Would it be possible for this data to be combined in real time with the NAO's picture of an area to make pathfinding and pattern-recognition faster? For instance, if a series of sensors were placed in each room of a building, the NAO could be integrated into a network with them. As the robot moved from room to room, it would receive data from its respective 'remote eye' in that room, thereby creating much more complete grids or pointclouds of a room, faster. This is another ambitious project because of the technical difficulties in networking and combining data from a myriad of sensors in an efficient manner. Its difficultly though could be matched by the value of studying data integration across remote sensors and what we would learn from such a project.

Using the NAO alone, it would also be valuable to study its ability to autonomously map and find paths through a new environment. However, this may only be possible with extra sensors (Kinect or LIDAR) on the NAO, as its two CMOS cameras are designed primarily for image and face recognition instead of creating pointclouds or mapping rooms. This project would evaluate the ability of a simple humanoid to gather data on an environment, including the rate and accuracy with which it could be acquired. Furthermore, it would be a chance to explore new ways of autonomous pathfinding with humanoids and integrating other sensors into the NAO.

Of course, there is an almost infinite set of other research projects that the NAO platform could facilitate, and many ideas for those will come from the SATE 2012 Air Force Lab's specifications. However, the above projects are interests of ourselves and the robotics research community, offering value to not only the Air Force Research Lab but robotics at large if they are executed well. Regardless, we look forward to discussing project ideas with the SATE students and advisers, and hope these can be included in the discussion.

0.4 References and Resources

In order of reference:

Aldebaran NAO Software - http://www.aldebaran-robotics.com/en/Discover-NAO/Software/choregraphe.html
Choregraphe documentation and tutorials - http://users.aldebaran-robotics.com/docs/site$_e$*n/green*
NAOROSStack − http : //www.ros.org/wiki/Robots/Nao
NAOqiFrameworkOverview−http : //www.aldebaran−robotics.com/documentation/dev/naoqi/
framework − overview
NaoQiAPI−http : //users.aldebaran−robotics.com/docs/site$_e$n/bluedoc/naoqi.html
−http : //www.aldebaran−robotics.com/documentation/naoqi/index.html
NAOSDKAPI−http : //www.aldebaran−robotics.com/documentation/dev/sdk.htmlsdk−
index
NAOFlashCardApplication − http : //www.about − robots.com/flash −
cards − on − nao.html
NAOConnect4Demonstration−http : //www.generationrobots.com/site/program−
nao − robot/index$_e$n.html
*NAOPlayingRock, Paper, Scissors, Lizard, Spock−http : //carlitoscontraptions.com/*2011/10/*n*
1337 − *plays − rock − paper − scissors − lizard − spock − at − maker −*
faire − ny − 2011/
*NAOBrushesaCat−http : //developer.aldebaran−robotics.com/blog/*2012/1/*nao−*
kinect − cat − a − winning − combination/
*NAORemotelyControlledviaKinect−http : //www.robots−dreams.com/*2011/01/*humanoid−*
robot−navigation−teleoperation−using−nao−and−kinect−video.html
*NAONEST−http : //singularityhub.com/*2011/10/11/*nao−robots−can−*
feed − themselves − by − plugging − into − their − nest/
CloudRobotics −
*UniFreiburg′sOsirisNAOforHumanoidRobotLocalizationinComplex*3*Denvironments−*
*http : //www.ros.org/news/*2010/10/*robots−using−ros−uni−freiburgs−*
osiris − nao.html
NAORobocupLeague−http : //www.aldebaran−robotics.com/en/Solutions/For−
Robocup/the − NAO − league.html
UniofFreiburg′sNAOROSPackages−http : //www.ros.org/wiki/alufr−
ros − pkg