Social Interactivity Mentor for Youth with Autism using the NAO Robot (SIMYAN)

Members: Andrew Nguyen, Bryce George, Colton Homuth, William Ross

Sponsor: Dr. Adham Atyabi

Advisor: Bill Michael

ECE 4890 / CE 4899 -001

May 10, 2021

Table of Contents

Problem Overview - written by William Ross

According to the American Psychiatric Association, "Autism spectrum disorder (ASD) is a complex developmental condition that involves persistent challenges in social interaction, speech and nonverbal communication, and restricted/repetitive behaviors. The effects of ASD and the severity of symptoms are different in each person" (Copeland, 2018). ASD is a relatively common condition in humans that can negatively affect social behavior and development. Children, especially those with ASD may have trouble developing the social skills necessary to find success is our social world. Thus, one proposed solution is the use of "social robotics" or robots aimed towards the development of social skills in humans. Our sponsor's academic research focus is on children with ASD.

Problem Statement - written by William Ross

- Solution should extend the NAOqi Framework SDK.
- Solution should enable the NAO to initiate an interactive drawing module with social feedback.
- The NAO must be able to draw simple shapes (circle, triangle, square, and rectangle) using a drawing implement on a writing surface.
- The NAO should verbally interact with the user before, during, and after the drawing sequence.
- The solution should include the integration of an external GPU to the NAO for faster on device processing using trained machine learning models.

Standards Discussion - written by Andrew Nguyen

Coding standards: Standard Python programming. Python code will be written in a PEP 8 formatting style. Each line of code will include 4 spaces instead of a tab. Comments will also be written per function call or block of code in doc-string format. Comments will describe all functions and its parameters. Comments will provide references/inks to documentation when available. For a possible custom utility is built, provide a command line or GUI interface.

Documentation Standards: For each code committed to the GitHub directory, UCCS-Social-Robotics/docs, it must include a Wiki page or markdown for each module. The codes must also include author(s), GitHub username, and date information at the top of documentation files and Wiki pages.

Safety Standards: IEC 60601 international standards for the basic safety and essential performance of medical electrical equipment. Provide safe interactions between the NAO robot and the user.

Constraints Discussion - written by Colton Homuth

Economic
The largest constraint in this case would be the budget. We need to acquire a depth sensor which might be lower quality than desired with the current budget, however this will not be a huge problem.

Environmental
For this project, we are not sure any environmental constraints would apply since this project takes place in a controlled, indoor environment without any other outside factors.

Social
Since this project is meant to promote social interaction within ASD children, it must be designed for this purpose. Mainly, we must avoid changing it explicitly into any particular therapeutic experience or anything else outside of our scope since we are not experts when it comes to working with ASD children. Moreover, no testing will be performed with ASD individuals at any point during this project.

Political
For this project, political forces should not be too much of a factor since all we are trying to do is create a safe, interactive space for ASD children.

Ethical
When dealing with ASD children, they must always be treated with respect and this would be our biggest ethical constraint. Since we're dealing with a user and one user alone, we must focus on them being treated well and not being used or mistreated in any way.

Health and Safety
For this project, since we are only dealing with one specific user, a big constraint would be making sure NAO programming does not prompt it to do anything dangerous. We must also consider the general closures and restrictions as a result of the COVID-19 pandemic, as well as the safety regulations for physically interacting with one another.

Manufacturability
Since project is adding software to existing hardware, we are only limited by the hardware on the NAO robot and the software we write to program its behaviors.

Sustainability
For this project, we will not have any problems with sustainability, as long as nothing is changed on the NAO robot and Python version that we are running.

Requirements Analysis and Literature Search - written by Bryce George

Customer Survey

We are working with a knowledgeable, frontier customer who has significant experience in the emerging research field of social robotics, and in particular, for their use as therapeutic or diagnostic tools working with individuals with ASD. His knowledge of the problem space, including the details of ASD manifestation, diagnosis, and treatment is considerable, and provides a solid foundational resource for verifying the efficacy, integrity, and applicability of a developed solution. That said, the field is still relatively nascent and there is a considerable degree to which the development of our solution will require either new techniques or novel applications of existing ones.

Literature Search

Our literature search of the problem space included a survey of the capabilities and applications of commercially available social robots specifically advertised for use with ASD individuals, and an exploration of the existing research on using social robotics for ASD diagnosis and therapy. During our commercial robot survey, we identified the robots listed in Table 1.

Table 1: Commercial Robots for ASD Research and Therapy

Robot	Made By	Height (cm)	Ambulatory	Programmable	Degrees of Freedom	Features
Milo	Robokind	~58	Yes	Yes	21	• Humanoid • High degrees of freedom allow for body language expression • Emotive verbal expression
NAO	SoftBank Robotics	58	Yes	Yes	25	
Pepper	SoftBank Robotics	120	No (wheels)	Yes	20	
QTrobot	LuxAI	63	No (stationary)	Yes	17	

The robots share a number of common features. With the exception of the Pepper robot, they all stand approximately two feet tall, making them about the same size as a human toddler. Most of them are able to move around, either by walking or using wheels, and they all exhibit high degrees of freedom. Their humanoid designs, ability to speak, and capacity to imitate human emotion all clearly establish their suitability for the purpose of serving as social robots.

From the research survey, we gained perspective on how social robots have been used for the purpose of treating or diagnosing ASD, as well as where opportunities for future innovations exist. Even within the research community, there is still a degree of controversy with regards to a number of issues, the most critical being: the appropriate role for robots within therapeutic settings, the effectiveness of such methods of treatment, and the socio-ethical dilemma of whether humans should be learning human behaviors from robots at all. In general, however, most agree that social robots have distinct advantages which lend themselves to certain aspects of treatment.

Human "[s]ocial-communication is a complex task that involves speech, gestures, facial expressions, and context. . . which are integrated automatically," into every communication within a social interaction (Diehl 2012). Many of the communication struggles for ASD individuals arise from an inability to integrate or prioritize social stimuli effectively, often leading to disengagement, over-stimulation, or distress. Unfortunately for human therapists and interactors, it is virtually impossible for a human to limit or eliminate these stimuli during treatment exercises, decreasing their effectiveness. On the other hand, the consistency with which social robots speak, move, and interact—as well as their more limited set of channels for delivering social signals—simplify social interactions, helping them to avoid such outcomes. A 2018 study conducted by LuxAI using their QTrobot found that ASD "children directed their gaze toward the robot about twice as long, on average, compared with their gaze towards the human," and repetitive stress behaviors, "occurred about three times as often during sessions with the human, compared with the robot" (Waltz 2018). These findings suggest that social robots may be more effective for engaging with ASD individuals in certain situations than human interactors.

One of the primary functions a social robot could provide is the opportunity to act as the "model social agent" whose, "goal would be to teach a skill that . . . the child could imitate or learn and eventually transfer to interactions with humans" (Diehl 2012). Within this capacity, the robot is advantageously suited to performing repetitive exercises, allowing the opportunity for a child to repeatedly practice behaviors, "without the social pressures of a peer interaction" (Diehl 2012). In a study conducted by Zhang et. al. and published in April of 2019, the researchers set out to evaluate the potential for a social behavior to be taught by a robot to a child, which might later be transferred to inter-human interactions. In addition, they sought to gain a better understanding on how social robots are ontologically viewed by children. In the study, children performed two activities with a NAO robot, wherein the robot attempted to instill critical social lessons related to distrust and deception (Zhang 2019). The results showed that, while ASD children still struggled more with learning distrust and deception concepts from the robot than their typically developing peers, they showed similar learning trends over the course of the trials, indicating that such learning could ultimately prove effective. With regards to children's perceptions of the robots, their finding suggest that humanoid robots are viewed as a unique anthropomorphic entity, with some degree of human-like characteristics and autonomy, but not fully human in terms of animacy and independence. In this context, children "treated them not only as social companions . . . but also as knowledgeable and informative interlocutors" (Zhang 2019). While the research study was not conducted to measure the efficacy of social robotic intervention with ASD individuals, it may still be regarded as a proof-of-concept study, demonstrating the potential for learning human behaviors from social robots.

Environment Survey

Since our customer made development of the solution on the NAO robot a requirement, we conducted a survey of the robot's capabilities as well as some of its potential shortcomings. Overall, the robot is quite versatile, offering 25 degrees of freedom and an open, extensible software development framework. The NAOqi Python Software Development Kit (SDK) provides a number of built-in modules to handle both basic and advanced operations related to vision, movement, speech, cognition, emotional detection and expression, human-like behaviors, and more. However, many of these modules have limitations which will necessitate the implementation of custom modules for the successful development of the proposed solution. Despite the NAOqi framework being open, its primarily proprietary development raises some concerns over the thoroughness of its documentation and comprehensiveness of its capabilities.

Beyond the robot and the NAOqi framework, we also took a look at the development community using the NAO robot. Of most significant note, we found 1) that the current NAO community is relatively small, and 2) that the community seems to be diminishing rather than growing. In terms of this project, that means we are unlikely to find existing community implementations for many of the advanced behaviors we will be looking to implement, and also, that we may experience difficulty in getting questions answered regarding the robot or its software that are not provided by SoftBank's documentation. Moreover, the community website for the NAO robot, hosted by SoftBank Robotics, appears to have been recently taken down, signaling an overall decline in community interest and engagement. These factors may pose a challenge during the project development stage.

Customer Needs and Wants

The customer stipulated the following needs during our first project planning meeting:

- A solution that extends the existing NAO framework
- Implementation of advanced social robotic behaviors in the following areas
    - Cognition
    - Speech
    - Vision
    - Mobility and Motor Control
- A behavioral exercise between the robot and a human individual providing a "play" experience
- The behavioral exercise must provide
    - Opportunity for the individual to learn social cues and concepts
    - Opportunity for the individual to perceive human mannerisms and social characteristics demonstrated by the robot

In addition to the above needs, the customer identified the following wants:

- Relatively autonomous operation of the robot during the behavioral exercise
    - Display intelligence
    - Display situational adaptability
- Provide a natural interaction experience between the robot and the human individual
    - Intuitive
    - Instructive
    - Comfortable
- Extensibility and reusability of the developed solution for future development

Project Scope

Given the team's limited medical knowledge of ASD and the techniques for its diagnosis or treatment, the following points remain out of scope of the current project:

- Development of an ASD diagnostic test
- Development of a therapeutic experience for ASD individuals
- Design or execution of a particular ASD-related research experiment
- Testing of a developed solution with any ASD individuals

Within the scope of the current project are the following objectives:

- Creation of a framework to aid the development and implementation of advanced social robotic behaviors to support diagnostic and therapeutic applications
- Development of a proof-of-concept behavior component to implement the specified social interactive drawing scenario

Requirements Analysis

In order to develop our requirements specifications, we began by defining an interaction scenario between the robot and a human subject. Based on our customer's needs and recommendations, we developed a social interactive drawing exercise, as well as identified a number of extensions to the foundational behaviors. With this scenario in mind, we approached our requirements specifications from a use case standpoint, identifying the primary functions the robot would need to be capable of performing in order to fulfill the scenario. These use cases encompass both the expected behaviors and the functional and quality of service requirements associated with them.

During the definition of the scenario use cases, we identified two primary interactive stages. The master-level stage is comprised of the actions necessary to initialize the robot, identify and make first contact with the human subject, elicit an activity selection from the subject, and start the desired activity. However, modules within this stage would remain responsible for monitoring for and executing high-level commands (such as ending an activity prematurely), as well as orchestrating the natural transition from one activity into another, throughout the duration of a session. These responsibilities embody the high-level behaviors we would expect to be present in almost any scenario where the robot is prepared to interact with an individual through some activity. As such, these use cases (Use Cases 1-3, 7, and 8) provide the specifications for the core functions of our developed framework, in addition to a number of utility modules to support common implementation needs at any level. The subordinate stage encompasses the actual execution of some activity involving the robot and the subject. For the current project, this stage is embodied by the execution of the social interactive drawing exercise, and its associated use cases are set forth in our requirements specification (Use Cases 4-6).

Requirements Specification - written by All

#	Use Case Name*	Functional Requirements	Quality of Service Requirements
1	Load Master Module	1. Allocate all necessary resources and load the Activities Master Module 2. Detect subject within 5m of starting position	1. Performed on robot startup 2. Verbal notification of completion
2	Initialize Activities Master Module	1. Verbally greet subject 2. Orient face to look at subject (+/- 5°) 3. Allocate/initialize all master level subroutines	1. Greet subject < 1s after detection
3	Start Activity	1. Verbally explain operation instructions 2. Verify valid verbal activity selections 3. Notify of invalid activity selections and re-prompt 4. Discover all available activity modules 5. Load and initialize selected activity module	1. Verify and acknowledge selections <= 1s after heard
4	Select Object to Draw	1. Verbally explain drawing activity 2. Verbally list drawable objects 3. Prompt for object selection 4. Verify object selection 5. Notify of invalid object selections and re-prompt	1. Verify and acknowledge selections <= 1s after heard
5	Draw Object	1. Locate/detect drawing surface within 5m of position 2. Notify of no drawing surface found and prompt for next action 3. Detect drawing surface boundaries 4. Notify inaccessible drawing surface conditions and instruct how to position correctly 5. Load necessary hand/arm motor control module(s) 6. Locate object drawing instructions 7. Execute drawing instructions to draw the object on the drawing surface	1. Detect drawing surface in <= 3s 2. Identify drawing surface boundaries in <= 1s

		8. Able to draw square a. Four angles within +/- 1° of 90° b. Four sides same length to within +/- 5mm 9. Able to draw rectangle a. Four angles within +/- 1° of 90° b. Two sets of sides same length to within +/- 5mm 10. Able to draw circle a. No segment of circumference varies by more than +/- 5% of radius length in distance from the center 11. Able to draw right triangle a. One angle +/-1° of 90° b. Two angles +/-1° of 45° c. Length of sides adjacent to 90° angle are same length within +/- 5mm	
6	Interact with Subject	1. Select an appropriate user interaction statement 2. Speak interaction statement to subject 3. Listen for verbal response to an interrogative statement 4. Evaluate response statement	1. Select or generate an interaction statement in <= 3s 2. Evaluate a response statement in <= 3s
7	Complete Activity	1. Evaluate whether activity can be repeated 2. Ask whether to repeat activity or exit activity 3. Verify and verbally acknowledge choice to repeat or exit the activity 4. Re-initialize activity module 5. Exit an unload activity module	1. Verify and acknowledge selections <= 1s after heard

** See the **Use Cases** section of the **Appendix** for use case definitions.*

Team Constraints - written by Bryce George

Andrew Nguyen is a Data Analytics and Systems Engineering major. He has skills in Python and software engineering. He is pursuing a job in Data Analytics. Limited knowledge in machine/deep learning.

Bryce George is pursuing B.S. degrees in Computer Science as well as Data Analytics and Systems Engineering and is also employed as a Software Engineer at Eclypses, a cyber security company located in Colorado Springs. He has considerable background in the .NET ecosystem as well as experience with the software engineering process, agile development, software security and computational cryptography, applied deep learning methods for object detection and tracking, IoT systems, and some exposure to robotics.

Colton Homuth is a 4th year Computer Engineering student mostly invested in software engineering. His plan is to take a software engineering job after graduation. In terms of this project, the area where he is most limited would be machine learning however he is excited to learn.

William Ross is an electrical engineering major in his 5th year. He is interested in software engineering and has some experience with python and deep learning research.

Overall, our team has limited experience with both Python and robotics, which will certainly be key areas of focused research and improvement for all team members. Several team members also have limited exposure to machine learning algorithms and the processes required for their development and application. In addition, developing familiarity and understanding of the proprietary NAO Framework will require a significant investment from the team and will be necessary in order to understand both its native capabilities, as well as how to implement the advanced social behaviors planned for this project. Finally, we will need to rely on the input of our sponsor and our own research to design the robot's behaviors and interactions to be as natural and effective as possible when working with ASD children.

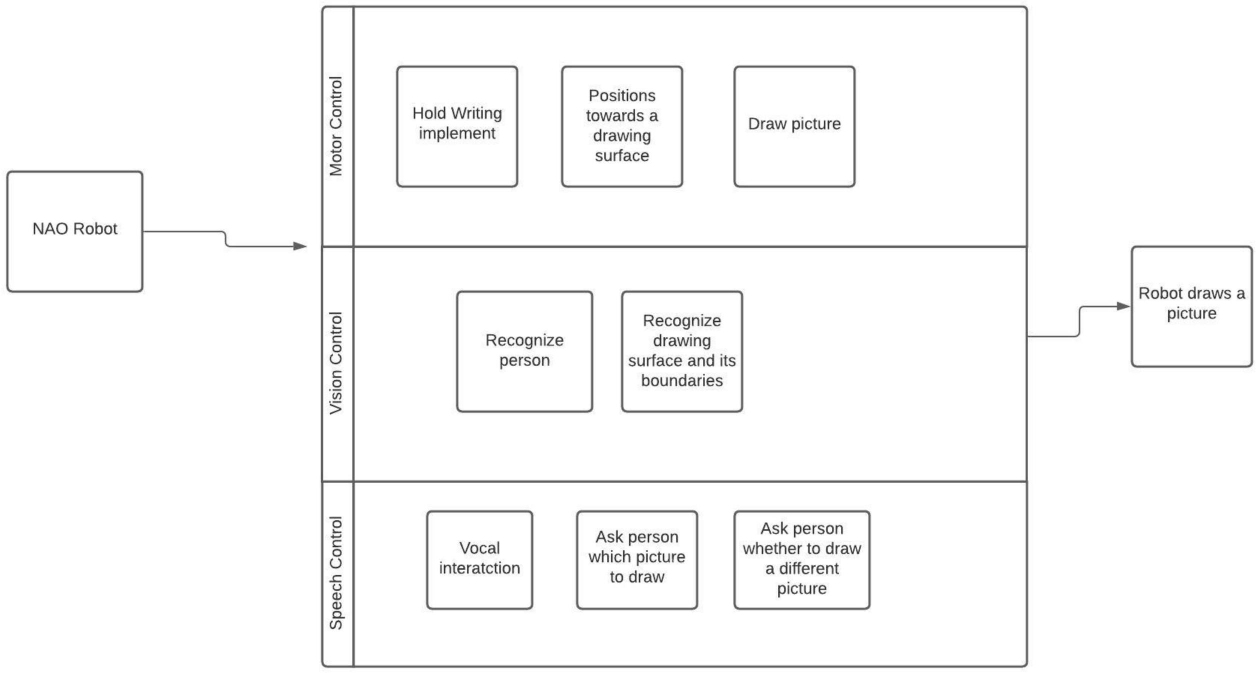
Operational Description - written by Bryce George

In order to go through the drawing demonstration activity with the NAO robot, the activity environment should be setup prior to initiating the activity. Setup includes securely attaching a dry erase marker in the hand of the robot, using tape or some mechanism to ensure the marker will not fall out of its hand or shift in its grip while drawing. The robot should then be placed in front of the drawing surface (white board) at a 12 inch distance, measured from the drawing surface plane to the back of the robot's foot. At this point, the Jetson Nano may be turned on (if being used), and then the robot turned on once the Jetson Nano has finished starting up. With the robot connected to the same network as the machine which will execute the demonstration code (which can be either the Jetson Nano or a computer connected to the same network), the demonstration can either be started using a runner provided by a Python IDE or run from the command line. To run from the command line, navigate to the `<path-to-repository>/src/py/drawing_demo` directory in the project repository and run:

```
$ python demo.py
```

Output logs from the activity should begin printing. Once the log message "Listening for trigger phrase…" is printed, the activity can be started. To start, the user should say "Let's draw something," to prompt the robot to begin the activity. The robot will then explain the activity and prompt the user to make a selection of what object to draw. The user should follow the robot's prompts to select a drawing. Once selected, the robot will begin drawing the requested object on the drawing surface. After completing the drawing, the robot will ask whether to repeat the activity and draw something else. The user may either answer affirmatively, in which case the robot will request for the user to clean the drawing surface and then prompt a new drawing selection, or negatively, to which the robot will verbally confirm that the activity is over.

Block Diagram - created by Andrew Nguyen


NAO Robot
Motor Control
Hold Writing implement
Positions towards a drawing surface
Draw picture
Vision Control
Recognize person
Recognize drawing surface and its boundaries
Speech Control
Vocal interatction
Ask person which picture to draw
Ask person whether to draw a different picture
Robot draws a picture

System Design Expectations - written by Colton Homuth

This project consisted of two main elements to be designed. The first was the SIMYAN core SDK which was meant to generally extend the vision, speech recognition, and motion capabilities of the NAO robot. The second was a drawing scenario which required the NAO to draw a picture on a whiteboard while interacting with another individual. The core SDK was meant to be a very general endeavor simply extending the existing NAO functionalities while the drawing scenario was meant to be an example showing what the core SDK could do. Originally, the biggest unknown was how to interact with the existing NAO functions and how to use the pre-built capabilities and this was our biggest struggle with the core SDK. After this, all that was needed was implementing the desired logic in the format the NAO SDK required. The most important aspect of the drawing scenario was the creation of a drawing algorithm able to make use of a predefined motor control module to make precise movements. Along with this, the NAO needed to be able to hold a writing utensil (dry erase marker), respond to verbal commands, recognize a writing surface (whiteboard), as well as interact with another person throughout the activity. The drawing activity was able to leverage many of the capabilities created in the core SDK along with leveraging some existing capabilities in the NAO SDK as needed.

Final Budget - written by William Ross

Project funding will be a total of $400 provided by the ECE department.

Items:

- Jetson Nano GPU - Free
- Realtek RTL8187L Chipset 2000mW Wireless USB Wifi Adapter 54Mbps Card - $24
- XBox Kinect - $50
- Power Cord - $7
- Kinect Adapters - $22
- Rubber Washers - $5

Total: $108

Deliverables - written by Bryce George

The primary deliverable for this project is the software SDK developed to enable a NAO robot to perform more advanced social behaviors. The custom SDK contains modules that provide the generic support components for implementing advanced social behaviors in the NAO robot, as well as a drawing demonstration that implements the SDK in order to exhibit its features and usage. Software documentation will also be delivered to explain the code and support future developments, as well as a manual for integrating and utilizing a Jetson Nano GPU within the NAO Framework.

- Software
  - SIMYAN SDK
  - SIMYAN SDK Documentation
  - Drawing Demo
  - Drawing Demo Documentation
- Jetson Nano GPU Integration Manual

Appendix - written by All

A. Problem Statement

Develop upon the NAO robot framework to provide capabilities for performing playful learning activities with human subjects. The activities should provide opportunities for the subject to learn human social skills and expression through interaction with the robot. The primary target of a developed solution would be children with ASD, and the ultimate goal would be to provide tools to help develop activities in the future which aid them in developing their social interactive capacity.

B. Specifications

Original Use Cases*

Acronyms:

AMM = Activities Master Module
SIDM = Social Interactive Drawing Module

** These use cases do not reflect the exact state of the final SDK and demonstration code's capabilities due to project re-scoping and limitations of the qi/NAOqi Framework discovered during development. They do provide a good overview of how a complex activity might be implemented and managed and provide indication of why certain decisions were made while developing the SIMYAN SDK and its service APIs.*

Use Case 1: Load Master Module (Bryce)
Actors: Robot, Operator
Entry Conditions: The Robot is available in the correct environment.
Exit Conditions: The master module has been loaded.

Flow of Events:

1) Initiate **Robot Startup**
2) Start to **Load Activities Master Module**
3) Prepare to **Detect Subject**

Exceptions:

2a) The AMM cannot be loaded
3a) The Robot cannot detect a subject within the allotted time

Use Case 1.1: Robot Startup (Bryce)
Actors: Robot, Operator
Entry Conditions: The Robot is available in the correct environment.
Exit Conditions: The Robot has completed its startup sequence.

Flow of Events:

1) The NAO button is pressed by the Operator
2) The Robot turns on

3) The Robot performs its standard startup sequence

Exceptions:

2a) The Robot does not turn on
3a) The Robot performs its software update sequence
- Action: Wait for the software update to complete and normal startup to resume.

3b) The Robot does not perform its standard startup sequence

Use Case 1.2: Load Activities Master Module (Bryce)
Actors: Robot, Operator
Entry Conditions: The Robot has been turned on and completed its startup sequence.
Exit Conditions: The AMM has been loaded.

Flow of Events:

1) The NAO system loads the AMM
2) The AMM self-initialization sequence is started
3) The Robot gives a verbal notification that the module has been loaded and initialized (**Speak Interaction Statement**)
4) The Robot pauses to allow the Operator to position it and leave the environment (assuming Operator is not Subject)
5) The AMM awaits the detection of a subject

Exceptions:

1a) The NAO system fails to load the AMM
1b) The NAO system cannot locate the AMM
2a) The AMM fails during its self-initialization sequence

Use Case 1.3: Detect Subject (Bryce)
Actors: Robot, Subject
Entry Conditions: The activities master module has been loaded and its initial sequence begun.
Exit Conditions: The Subject has been detected.

Flow of Events:

1) The Subject enters the environment
2) The Robot detects the Subject using its camera and audio systems
3) The Robot turns its head to look at the Subject
4) An event is raised to notify the AMM that the Subject has been detected

Exceptions:

1a) The Subject does not enter the environment in the allotted amount of time
2a) The Robot fails to detect a Subject within the environment
3a) The Robot does not turn its head to look at the Subject
3b) The Robot turns its head, but does not look at the Subject directly

<u>Use Case 2: Initialize Activities Master Module (Bryce)</u>
Actors: Robot, Subject
Entry Conditions: The AMM has been loaded and the Subject has been detected.
Exit Conditions: The Subject has been greeted and all activity modules have been discovered.

Flow of Events:

1) The AMM receives the Subject detected event
2) The Robot greets the Subject (**Greet Subject**)
3) The AMM **Discovers Available Activity Modules**

Exceptions:

1a) The AMM does not receive the Subject detected event in the allotted time
3a) The Robot cannot locate any available Activity Modules

Use Case 2.1: Greet Subject (Bryce)
Actors: Robot, Subject
Entry Conditions: The Subject has been detected.
Exit Conditions: The Subject has been greeted.

Flow of Events:

1) Select a greeting (**Select Interaction Statement**)
2) Speak greeting to the Subject (**Speak Interaction Statement**)
3) Invite the Subject to approach (**Speak Interaction Statement**)

Exceptions:

N/A

Use Case 2.2: Discover Available Activity Modules (Bryce)
Actors: Robot
Entry Conditions: The activities master module has been loaded.
Exit Conditions: All activity modules have been discovered.

Flow of Events:

1) The AMM scans for Activity Modules
2) The AMM loads the descriptions for each module found

Exceptions:

1a) The AMM cannot locate any Activity Modules
2a) An Activity Module is missing a description

Use Case 3: Start Activity (Bryce)
Actors: Robot, Subject
Entry Conditions: The activities master module has been initialized.
Exit Conditions: An activity module has been selected, loaded, and its initial operation sequence begun.

Flow of Events:
1) The Robot **Explains Commands**
2) The Robot **Explains Activities**
3) The Robot allows the Subject to **Select Activity**
4) The Robot **Loads the Selected Activity Module**
5) The activity module initialization sequence is started (**Start Activity Module Initialization Sequence**)

Exceptions:
4a) An activity selection is not heard within the allotted time
5a) An invalid selection is heard
- Action: **Handle Invalid Activity Selection**

Use Case 3.1: Explain Commands (Bryce)
Actors: Robot, Subject
Entry Conditions: The activities master module has been initialized.
Exit Conditions: The available master-level commands have been explained to the subject.

Flow of Events:
1) Speak instructions for each master-level command (**Speak Interaction Statement**)

(Note: List of master-level commands to be defined)

Exceptions:
N/A

Use Case 3.2: Explain Activities (Bryce)
Actors: Robot, Subject
Entry Conditions: The activities master module has been initialized and all activity modules discovered.
Exit Conditions: The available activities modules have been explained.

Flow of Events:
1) Speak the name of each activity module and its description (**Speak Interaction Statement**)

Exceptions:
N/A

Use Case 3.3: Select Activity (Bryce)
Actors: Robot, Subject
Entry Conditions: The available activities have been explained to the subject.
Exit Conditions: The subject has selected a valid activity.

Flow of Events:

1) Prompt the subject to speak the name of an activity (**Speak Interaction Statement**)
2) **Listen for Selection** of an activity
3) The Robot **Validates the Activity Selection**

Exceptions:

3a) The Subject selects an invalid activity
- Action: **Handle Invalid Activity Selection**

3b) The Subject does not select an activity within the allotted time

Use Case 3.4: Listen for Selection (Bryce)
Actors: Robot, Subject
Entry Conditions: The activities master module has been initialized, the Subject identified, and the activity modules list spoken to the Subject.
Exit Conditions: The Robot has heard a potential selection or command.

Flow Events:

1) The Robot waits to hear a response from the Subject (**Listen for Question Response**)
2) The Robot forwards a complete statement to the appropriate modules for processing

Exceptions:
N/A

Use Case 3.5: Validate Activity Selection (Bryce)
Actors: Robot
Entry Conditions: The Robot has heard a potential activity selection.
Exit Conditions: The Robot has identified a potential activity selection as valid or invalid.

Flow of Events:

1) The Robot evaluates the Subject's statement
2) The Robot identifies a valid activity selection
3) The Robot forwards the activity selection to the AMM to be loaded

Exceptions:

2a) The Robot does not identify a valid activity selection
Action: **Handle Invalid Activity Selection**

Use Case 3.6: Handle Invalid Activity Selection (Bryce)
Actors: Robot, Subject
Entry Conditions: The robot has identified an invalid activity selection.
Exit Conditions: The robot has notified the user of the invalid selection and prompted for the next action to be taken.

Flow of Events:

1) The Robot verbally notifies the Subject that it could not identify a valid activity selection (**Speak Interaction Statement**)
2) The Robot asks the Subject whether they wish to hear the activity module list again (Speak Interaction Statement)
3) If the Subject wishes to hear the activity module list, the Robot **Explains Activities** again
4) The Robot prompts the Subject to **Select Activity**

Exceptions:
N/A

Use Case 3.7: Load Selected Activity Module (Bryce)
Actors: Robot
Entry Conditions: The robot has identified a valid activity selection.
Exit Conditions: The activity module has been loaded.

Flow Events:

1) The activity module is loaded
2) All necessary module resources are initialized/allocated

Exceptions:
N/A

Use Case 3.8: Start Activity Module Initialization Sequence (Bryce)
Actors: Robot
Entry Conditions: An activity module has been loaded.
Exit Conditions: The activity module's initialization sequence has begun.

Flow Events:

1) The module initialization sequence is started

Exceptions:
N/A

<u>Use Case 4: Select Object to Draw (Colton)</u>

Actors: Robot, Subject
Entry Conditions: The Social Interactive Drawing Module has been loaded and initialized.
Exit Conditions: A valid object has been selected by the subject for the robot to draw.

Flow Events:

1) Verbally explain drawing activity (**Speak Interaction Statement**)
2) The Robot **Lists Drawable Objects**
3) The Robot **Prompts the Subject to Select Object**
4) The Robot **Validates the Object Selection**

Exceptions:

4a) A selection is not heard by the robot
4b) An invalid selection is heard
- Action: **Handle Invalid Object Selection**

Use Case 4.1: List Drawable Objects (Colton)
Actors: Robot, Subject
Entry Conditions: The Social Interactive Drawing Module has been loaded and initialized.
Exit Conditions: The available objects that can be drawn have been listed to the subject.

Flow Events:

1) Scan for available drawable object instructions
2) List all available drawable objects to Subject (**Speech Interaction Statement**)

Exceptions:

Use Case 4.2: Prompt Object Selection (Colton)
Actors: Robot, Subject
Entry Conditions: The robot has listed all available objects which can be drawn to the subject.
Exit Conditions: The robot has prompted the subject to select an object to draw.

Flow Events:

1) Prompt the Subject to select one of the objects listed (**Speech Interaction Statement**)

Exceptions:

Use Case 4.3: Validate Object Selection (Colton)
Actors: Robot
Entry Conditions: The robot has heard a potential object selection.
Exit Conditions: The robot has identified whether the object selection is valid or invalid.

Flow Events:

1) Listen to the Subject state one of the objects listed
2) Make sure object that the Subject selected is one of available objects

Exceptions:

1a) A selection is not heard by the robot
2a) A selection is not one of valid selections
- Action: **Handle Invalid Object Selection**

Use Case 4.4: Handle Invalid Object Selection (Colton)
Actors: Robot, Subject
Entry Conditions: The robot has identified an invalid object selection.
Exit Conditions: The robot has notified the subject of the invalid selection and prompted for the next action to be taken.

Flow Events:

1) The Robot tells the Subject that their selection is invalid (**Speech Interaction Statement**)
2) The Robot **Prompts the Subject to Select Object**

Exceptions:

Use Case 5: Obtain Writing Implement (Andrew)
Actors: Robot, Subject
Entry Conditions: The robot has been instructed to draw a valid object.
Exit Conditions: The robot has obtained a writing implement which is oriented correctly for drawing.

Flow Events:

1) Robot asks Subject which writing implement to draw a picture (**Interact with Subject)**
2) Robot finds and picks up a writing implement
3) Robot verifies and confirms Subject's selection of writing implement
4) Robot draws a picture (**Draw Object**)

Exceptions:

2a) Robot cannot find a writing implement
2b) Robot cannot hold a writing implement (**Evaluate Marker Orientation**)

Use Case 5.1: Determine if Holding Marker (Andrew)
Actors: Robot
Entry Conditions: The robot has been instructed to draw a valid object.
Exit Conditions: The robot has determined whether it is already holding a marker or not.

Flow Events:

1) Robot scans the environment and finds a marker
2) Robot picks up the marker
3) Robot determines if it is holding a marker correctly (**Evaluate Marker Orientation)**

Exceptions:

1a) Robot scans environment and does not see a marker

Use Case 5.2: Ask Whether to Use Current Marker (Andrew)
Actors: Robot, Subject
Entry Conditions: The robot has determined that it is already holding a marker.
Exit Conditions: The robot has determined whether it should continue using the current marker or obtain a new one.

Flow Events:

1) Robot holds current marker from Subject's selection
2) Robot verifies and confirms current marker with Subject (**Validate Marker Selection**)
3) Robot draws a picture with the marker selected from the Subject

Exceptions:

2a) Subject changes marker selection (**Validate Marker Selection**)

Use Case 5.3: Validate Marker Selection (Andrew)
Actors: Robot
Entry Conditions: The robot has heard a potential marker selection instruction.
Exit Conditions: The robot has identified whether the selection instruction statement is valid or invalid.

Flow Events:

1) Robot listens to Subject's selection of marker to color the picture
2) Robot asks Subject which marker to color the picture (**Interact with Subjec**t)
3) Robot determines if Subject's selection of marker is valid
4) Robot finds and picks the marker selected from the Subject

Exceptions:

3a) Subject's choice of marker is invalid for the Robot

Use Case 5.4: Drop Marker (Andrew)
Actors: Robot
Entry Conditions: The robot needs to obtain a new marker but is currently holding one.
Exit Conditions: The robot is no longer holding the marker.

Flow Events:

1) Robot holds and draws with the current marker

2) Robot finishes drawing a picture with the current marker (**Complete Activity**)
3) Robot drops the marker on the floor

Exceptions:
1a) Robot cannot hold a new marker correctly (**Evaluate Marker Orientation)**
2a) Robot does not finish drawing with current marker (**Request New Marker**)

Use Case 5.5: Request New Marker (Andrew)
Actors: Robot, Subject
Entry Conditions: The robot has determined that it needs to obtain a new marker and is not currently holding a marker.
Exit Conditions: The robot has requested the subject to provide a new marker and the subject has placed the marker in its hand.

Flow Events:
1) Robot draws a picture with a marker selected from the Subject
2) Robot asks Subject to continue drawing with current marker (**Interact with Subject**)
3) Robot determines whether Subject wants to change current marker to a new marker

Exceptions:
3a) Subject does not want to change his/her selection of the current marker

Use Case 5.6: Evaluate Marker Orientation (Andrew)
Actors: Robot
Entry Conditions: The robot has a marker in its hand.
Exit Conditions: The robot has determined whether the marker is oriented correctly for drawing.

Flow Events:
1) Robot picks up a marker with one hand with a full-fist orientation
2) Robot determines if marker's orientation can be held by its hand

Exceptions:
2a) Marker's orientation cannot be hold by Robot's hand

Use Case 5.7: Evaluate Whether Cap On/Off (Andrew)
Actors: Robot
Entry Conditions: The robot has a marker in its hand which has been determined to be oriented correctly.
Exit Conditions: The robot has determined whether the marker's cap is on or off.

Flow Events:
1) Robot is able to hold a marker (**Evaluate Marker Orientation**)
2) Robot examines the marker

3) Robot determines if marker's cap is on or off (**Handle Marker Cap On**)

Exceptions:

3a) Marker's cap is already off, Robot does not need to examine the marker

Use Case 5.8: Handle Invalid Marker Orientation (Andrew)
Actors: Robot, Subject
Entry Conditions: The robot has determined that it is holding a marker in an incorrect orientation for drawing.
Exit Conditions: The robot has notified the subject that the marker is incorrectly oriented and prompted for the next action to be taken.

Flow Events:

1) Robot is not able to hold the marker correctly (**Evaluate Marker Orientation**)
2) Robot notifies Subject it cannot hold the marker correctly
3) Robot asks Subject to select a different marker to draw (**Interact with Subject**)

Exceptions:

2a) Robot can hold a marker correctly (**Evaluate Marker Orientation)**

Use Case 5.9: Handle Marker Cap On (Andrew)
Actors: Robot, Subject
Entry Conditions: The robot has determined that it is holding a marker with its cap on.
Exit Conditions: The robot has prompted the subject to remove the cap and the subject has done so.

Flow Events:

1) Robot can hold the marker upward
2) Robot examines and determine marker's cap is on or off (**Evaluate Whether Cap On/Off**)
3) Robot asks Subject to remove the marker's cap if it is on the marker

Exceptions:

2a) Marker's cap is already removed (**Evaluate Whether Cap On/Off)**
3a) How tight the marker is on for the Subject to remove

<u>Use Case 6: Draw Object (Will)</u>
Actors: Robot, Subject
Entry Conditions: The robot has been instructed to draw a valid object and is holding a writing implement properly for drawing.
Exit Conditions: The robot has drawn the requested object on the drawing surface.

Flow Events:

1) The robot identifies the drawing surface and boundaries.

2) The robot draws the selected picture on the drawing surface.
3) The robot interacts with the user in the event of any errors.

Exceptions:

1a) The robot cannot locate or access drawing surface or identify its boundaries.

Use Case 6.1: Locate Drawing Surface (Will)
Actors: Robot, Subject
Entry Conditions: The robot has been instructed to draw a valid object and is holding a writing implement properly for drawing.
Exit Conditions: The robot has located a drawing surface.

Flow Events:

1) The robot scans the environment for the drawing surface.

Exceptions:

1a) The robot does not find the drawing surface.

Use Case 6.2: Determine Drawing Surface Boundaries (Will)
Actors: Robot
Entry Conditions: The robot has located a drawing surface.
Exit Conditions: The robot has identified the boundaries of the drawing surface.

Flow Events:

1) The robot has located the drawing surface.
2) The robot scans the drawing surface to determine its boundaries.

Exceptions:

2a) The robot is unable to locate the boundaries.

Use Case 6.3: Evaluate Drawing Surface Accessibility (Will)
Actors: Robot
Entry Conditions: The robot has located a drawing surface and identified its boundaries.
Exit Conditions: The robot has evaluated whether a drawing surface is accessible or inaccessible.

Flow Events:

1) The robot has determined and stored the boundary locations of the drawing surface.
2) The robot determines if the surface is accessible.

Exceptions:

2a) The robot cannot access the drawing surface.

Use Case 6.4: Activate Advanced Motor Control Module (Will)
Actors: Robot
Entry Conditions: The robot has identified an accessible drawing surface.
Exit Conditions: The robot has activated the advanced motor control module.

Flow Events:
1) The robot has determined the drawing surface to be accessible.
2) The robot loads the module for advanced motor control.

Exceptions:
N/A

Use Case 6.5: Execute Object Drawing Instructions (Will)
Actors: Robot
Entry Conditions: The robot has been instructed to draw a valid object and has activated the advanced motor control module.
Exit Conditions: The robot has drawn the instructed object on the drawing surface.

Flow Events:
1) The robot has successfully loaded the advanced motor control module.
2) The robot executes the correct drawing instructions for the shape specified by the user previously.

Exceptions:
N/A

Use Case 6.6: Handle Inaccessible Drawing Surface (Will)
Actors: Robot, Subject
Entry Conditions: The robot has located a drawing surface but has determined that is inaccessible.
Exit Conditions: The robot has notified the subject of the inaccessible drawing surface and prompted the subject to position it properly.

Flow Events:
1) The robot determines the drawing surface to be inaccessible.
2) The robot notifies the user of an inaccessible drawing surface and prompts the user to make adjustments.
3) The robot waits for the user to make the surface accessible and give a command to resume or for the user to terminate the program.

Exceptions:
N/A

Use Case 6.7: Handle No Drawing Surface (Will)
Actors: Robot, Subject
Entry Conditions: The robot has been unable to locate a drawing surface in the environment.
Exit Conditions: The robot has notified the subject that it cannot locate a valid drawing surface and has prompted the subject to select the next action.

Flow Events:

1) The robot determines that it cannot find the drawing surface.
2) The robot notifies the user that it cannot find the drawing surface and prompts the user to make adjustments.
3) The robot waits for the user to make adjustments and give a command.

Exceptions:
N/A

<u>Use Case 7: Interact with Subject (Bryce)</u>
Actors: Robot, Subject
Entry Conditions: The robot has determined that it needs to interact with the subject.
Exit Conditions: The robot has completed the interaction with the subject.

Flow Events:

1) The Robot **Selects an Interaction Statement**
2) The Robot **Speaks the Interaction Statement**
3) If the statement was a question, the Robot **Listens for Question Response**

Exceptions:
N/A

Use Case 7.1: Select Interaction Statement (Bryce)
Actors: Robot
Entry Conditions: The robot has determined that it needs to speak to the subject.
Exit Conditions: The robot has selected an interaction statement to speak to the subject.

Flow Events:

1) Communication providers within the NAO system are polled to determine the appropriate handler
2) The appropriate handler assesses the environment and system state
3) The handler generates an interaction statement

Exceptions:
1a) No communication handler is available to handle the current situation

Use Case 7.2: Speak Interaction Statement (Bryce)
Actors: Robot, Subject
Entry Conditions: The robot has selected an interaction statement to speak to the subject.
Exit Conditions: The robot has spoken the interaction statement to the subject.

Flow Events:
1) The NAOqi Text-to-Speech module receives a posted instruction containing text to speak
2) The module processes the text and speaks the resulting message

Exceptions:
N/A

Use Case 7.3: Listen for Question Response (Bryce)
Actors: Robot, Subject
Entry Conditions: The robot has asked the subject a question.
Exit Conditions: The robot has heard a possible response to the question.

Flow Events:
1) The Robot's microphone's collect audio data
2) Collected audio data is processed to construct a linguistic statement
3) Identified linguistic statements are posted to the AMM communications broker
4) The AMM communications broker **Evaluates the Question Response**
5) A communication handler triggers the appropriate actions to handle the question response

Exceptions:
N/A

Use Case 7.4: Evaluate Question Response (Bryce)
Actors: Robot
Entry Conditions: The robot has heard a possible response to a question.
Exit Conditions: The robot has evaluated a subject's response to a question and determined the next action to take.

Flow Events:
1) The AMM communications broker assesses whether the statement is a master level command
2) If the AMM cannot identify a master level command, the statement is forwarded to each communication handler in order of most recent registration until the statement is handled

Exceptions:
2a) No communication handler is able to handle the statement

Use Case 8: Complete Activity (Bryce)
Actors: Robot, Subject
Entry Conditions: The robot has completed the sequence for an activity.
Exit Conditions: The robot has determined whether to exit or restart the activity.

Flow Events:
1) The Robot assesses whether an activity is repeatable
2) If the activity is not repeatable, the AMM **Exits the Activity Module**
3) Otherwise, the Robot **Prompts Whether to Continue the Activity**
4) If the Subject elects to continue, the AMM **Restarts the Activity Module**

Exceptions:
N/A

Use Case 8.1: Prompt Whether to Continue Activity (Bryce)
Actors: Robot, Subject
Entry Conditions: The Robot has recognized some condition indicating that an activity may need to be ended or restarted.
Exit Conditions: The robot has prompted the subject to select the next action.

Flow Events:
1) The Robot notifies the Subject that an activity may need to be ended (**Speak Interaction Statement**)
2) The Robot asks whether the Subject is ready to end the current activity (**Speak Interaction Statement**)
3) The Robot listens for the Subject's response (**Listen for Question Response**)
4) If the Subject has selected to end the activity, the AMM **Exits the Activity Module**, otherwise the activity is resumed
5) If the activity has been exited, the Robot prompts the Subject to **Select an Activity**

Exceptions:
2a) The current activity must be ended
- Action: **Exit Activity Module**

Use Case 8.2: Restart Activity Module (Bryce)
Actors: Robot
Entry Conditions: The robot has determined that an activity module needs to be restarted.
Exit Conditions: The robot has started the activity module's initialization sequence.

Flow Events:
1) The AMM clears the current activity state
2) The AMM runs the activity module's initialization sequence

Exceptions:

N/A

Use Case 8.3: Exit Activity Module (Bryce)
Actors: Robot
Entry Conditions: The robot has determined that it needs to exit the current activity module.
Exit Conditions: The robot has exited the current activity module and returned to the activity selection stage.

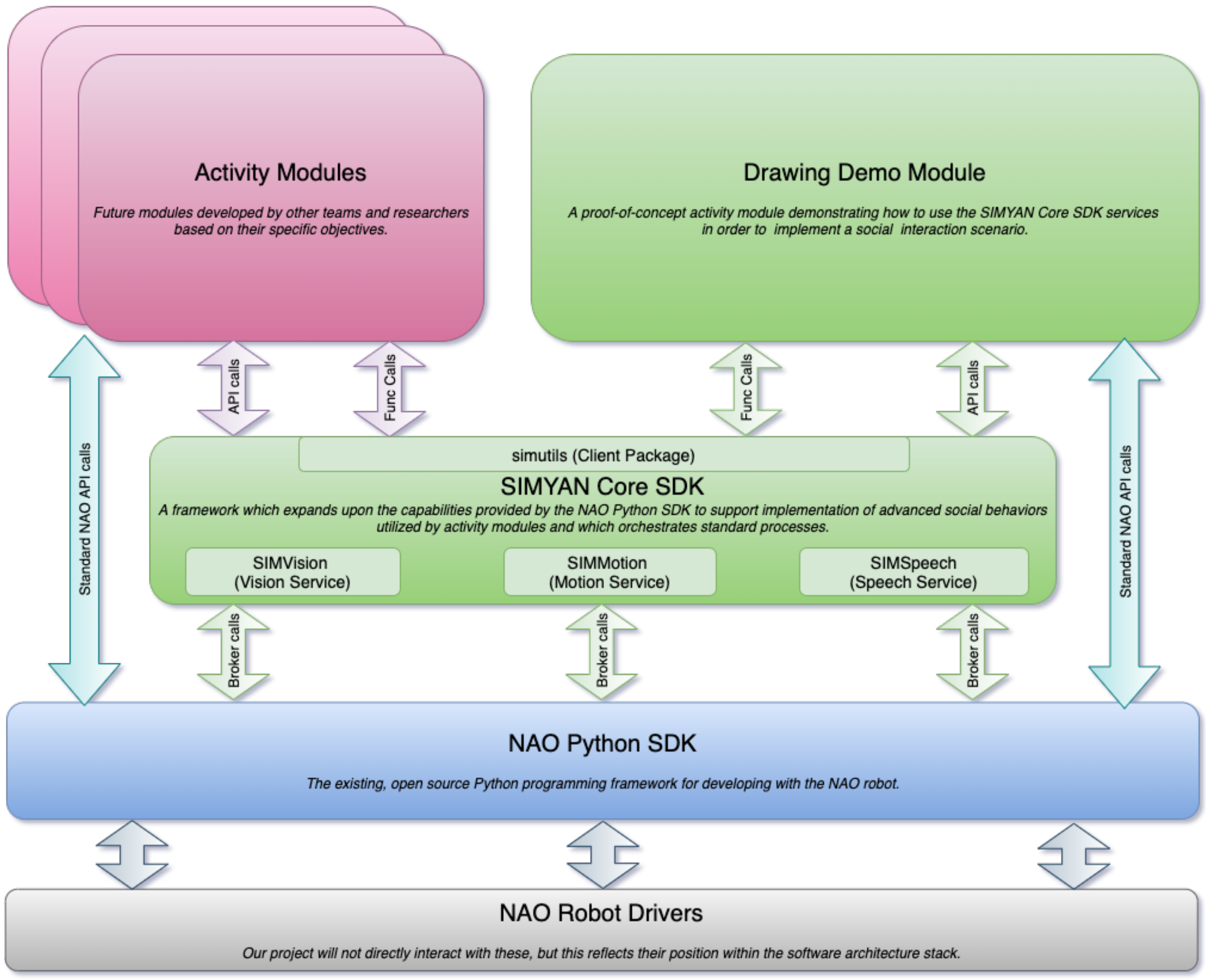
Flow Events:

1) The AMM clears the current activity state
2) The AMM unloads the activity module

Exceptions:
N/A

2. Schematics

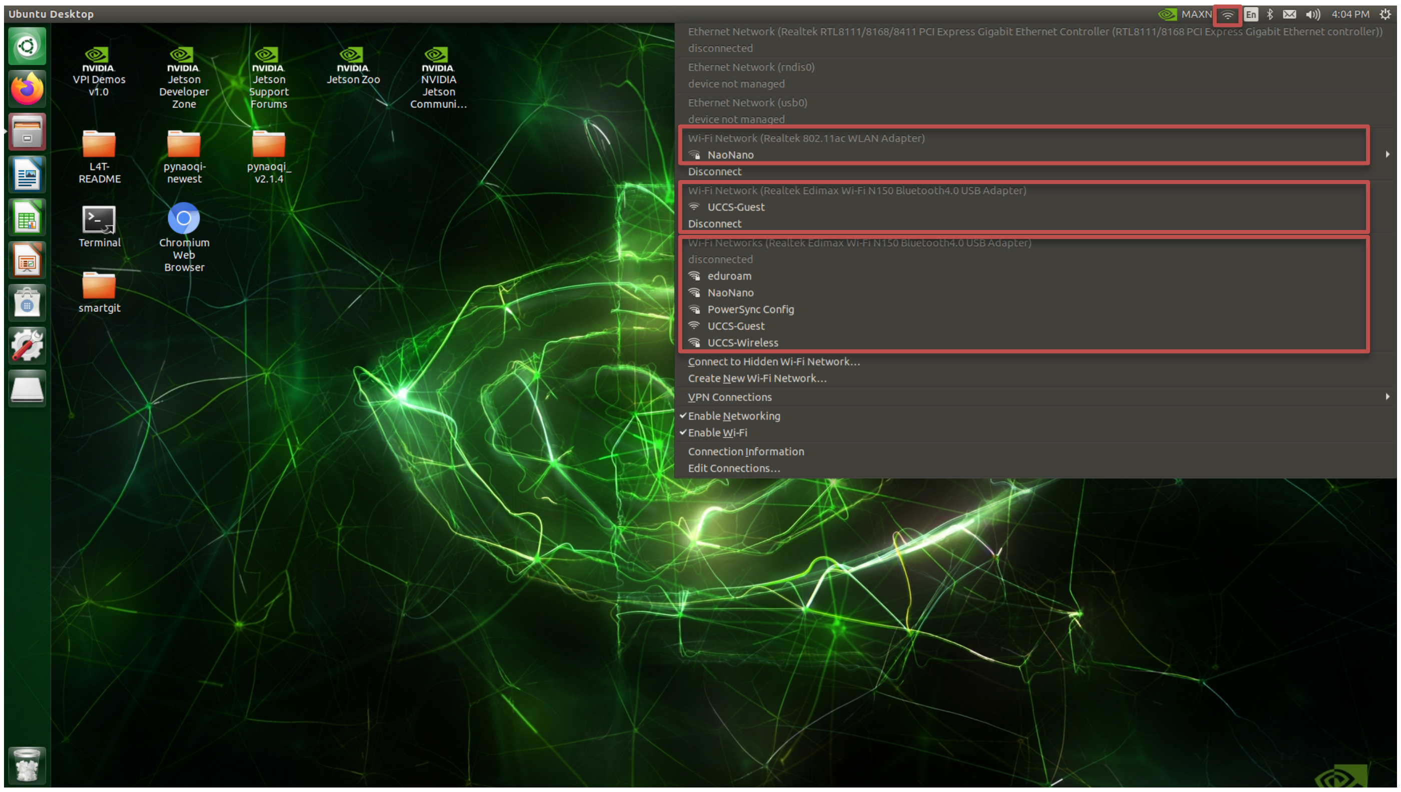
Software Architecture


Activity Modules
Future modules developed by other teams and researchers based on their specific objectives.
Drawing Demo Module
A proof-of-concept activity module demonstrating how to use the SIMYAN Core SDK services in order to implement a social interaction scenario.
Standard NAO API calls
API calls
Func Calls
Func Calls
API calls
Standard NAO API calls
simutils (Client Package)
SIMYAN Core SDK
A framework which expands upon the capabilities provided by the NAO Python SDK to support implementation of advanced social behaviors utilized by activity modules and which orchestrates standard processes.
SIMVision (Vision Service)
SIMMotion (Motion Service)
SIMSpeech (Speech Service)
Broker calls
Broker calls
Broker calls
NAO Python SDK
The existing, open source Python programming framework for developing with the NAO robot.
NAO Robot Drivers
Our project will not directly interact with these, but this reflects their position within the software architecture stack.

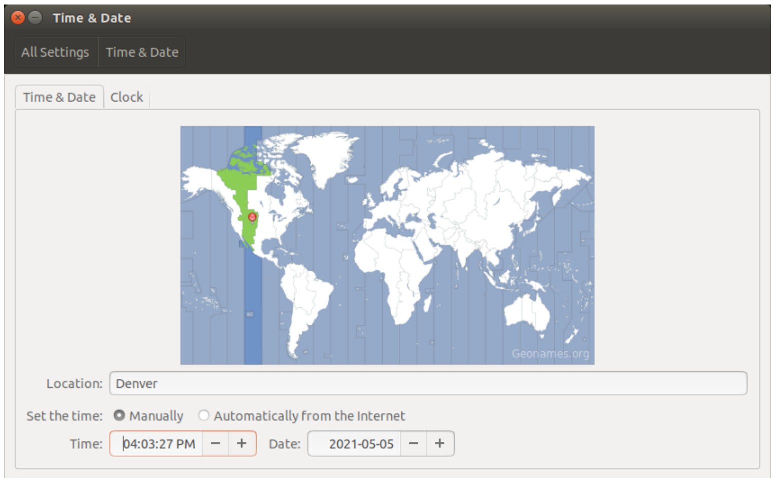
C. Manuals for Operation

i. Starting the Jetson Nano

1. Insert both the EDUP and Edimax WiFi adapters into USB ports on the Jetson Nano.
2. Connect the board to a keyboard, mouse, and display.
3. Plug the power cord into an outlet, then plug the other end into the board. The green LED on the board should begin flashing and Ubuntu should boot automatically.
4. Login to the 'uccs' account when the login screen displays.
5. Expand the network menu by clicking the icon at the far right of the menu bar. Three connections should show, one for the EDUP antenna and two for the Edimax. Leave the EDUP antenna connected the NaoNano network and disconnect the other two.
6. If connecting to another WiFi network for internet access, attempt to connect using the first Edimax antenna. If connection fails, try using the other one (in our experience, one works and the other does not, but it is not consistent as to which antenna works). Leave the one which does not work disconnected.



   a. If connecting to the UCCS Guest network, or another network with WPA2 authentication which requires a login/acceptance of terms, you will likely need to set the system time and date. This can be done by clicking on the time in the far-right corner of the main menu bar and selecting the 'Time & Date Settings" options.



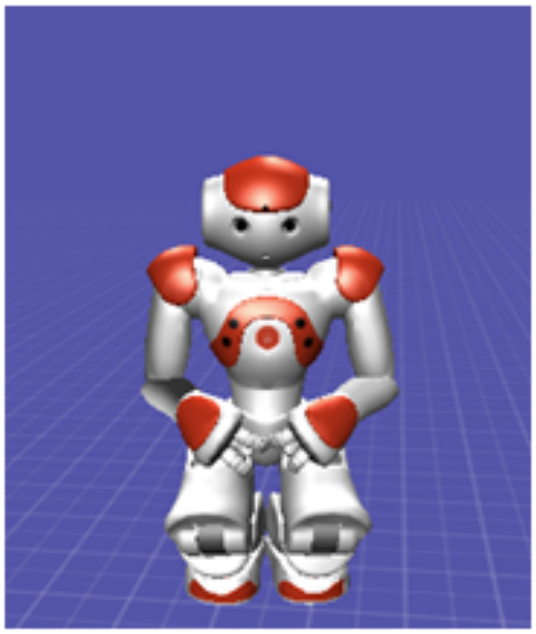
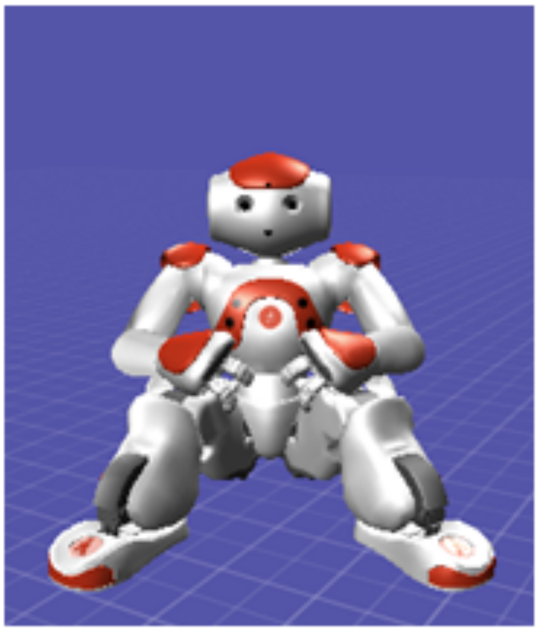
Location: Denver

Set the time: [✓ Manually] Automatically from the Internet

Time: 04:03:27 PM Date: 2021-05-05

ii. Starting the Robot

1. Remove the robot from its container and set it upon a flat surface in either a crouched or sitting position.

Crouch	Sit
	

2. Briefly press and then release the chest button. The robot's chest and eye LEDs will begin to go and may flash and change colors. Startup can be quick or may take several minutes. Always give at least 10-15 minutes of time for the robot to startup before attempting to hard boot or otherwise interrupt the startup sequence.

3. If the Jetson Nano is started and hosting the NaoNano network, the robot should connect automatically. If not, you will need to connect it to your desired network. (See additional instructions in the project repository documentation.)
4. Quickly press the robot's chest button once. The robot should say "Hi, I'm Nao. My IP address is <ip-address>." If the address begins with 192, then the robot is not connected to a network and you will need to plug an ethernet cable into the port at the back of its head in order to connect. If the robot is connected to the Jetson Nano, the address should be something like 10.0.X.XX.

iii. Executing the Drawing Demo Activity

1. Using tape or some other mechanism secure a marker in the robot's left hand.
2. Position the robot facing a whiteboard as a distance of 12 inches, measured from the whiteboard plane to the back of the robot's foot.
3. Having performed steps (i) and (ii), you can choose to either run the demo from the Jetson Nano or from another computer on the same network as the robot.
4. To run the demo from the command line, navigate to the ~/git/UCCS-Social-Robotics/src/py/drawing_demo directory (or the path to the equivalent location on your system). Then run:

   ```
   $ python demo.py
   ```

   Note: If you get an error saying that modules from the `simutils` *package cannot be found while attempting to run the demo, you will need to add the* `simutils` *package to your Python path variable or copy the* `simutils` *package into the* `drawing_demo` *directory.*

5. You can also run the demo by opening the script in a Python IDE and running it using the IDE's runner.
6. Once the demo has started, the activity logs should begin printing to the console. Once the log "Listening for trigger phrase…" has been printed, the activity is ready to begin.
7. Say "Let's draw something" loudly and clearly to the robot.
8. Follow the prompts from the robot to select a drawing and proceed through the activity.

See the project wiki for additional information, tutorials, and resources.

See the /src/py/simyan/app/scripts/DEVELOPERS.md file for useful SDK information.

D. Parts List

- NAO Robot
- Jetson Nano 4GB Developer Kit
- SMAKN DC 5V 2.5A 20W Switching Power Supply
- Edimax N150 Wi-Fi Bluetooth 4.0 USB Adapter (EW-7611ULB)
- EDUP AC Dual Band WiFi USB Adapter with 2DBI External Antenna (EP-AC1607)

- microSD Card (>= 32GB)
- Dry Erase Marker
- Whiteboard
- Duct Tape (Recommended)

E. Additional Information

- Project GitHub Repository: https://github.com/ancient-sentinel/UCCS-Social-Robotics
- Project Wiki: https://github.com/ancient-sentinel/UCCS-Social-Robotics/wiki
- UCCS GitHub Social Robotics Organization: https://github.com/UCCS-Social-Robotics
- Project Fork in UCCS Social Robotics: https://github.com/UCCS-Social-Robotics/simyan

Compiled by Bryce George

References

Copeland, J. N. (2018, August). What Is Autism Spectrum Disorder. Retrieved December 03, 2020, from https://www.psychiatry.org/patients-families/autism/what-is-autism-spectrum-disorder

Diehl, J., Schmitt, L., Villano, M., & Crowell, C. (2012, January). The Clinical Use of Robots for Individuals with Autism Spectrum Disorders: A Critical Review. Retrieved November 30, 2020, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3223958/

NAO^{6}. (n.d.). Retrieved Fall, 2020, from https://developer.softbankrobotics.com/nao6

Waltz, E. (n.d.). Therapy Robot Teaches Social Skills to Children With Autism. Retrieved November 28, 2020, from https://spectrum.ieee.org/the-human-os/biomedical/devices/robot-therapy-for-autism

Zhang, Y., Song, W., Tan, Z., Zhu, H., Wang, Y., Lam, C., . . . Yi, L. (2019, April 09). Could social robots facilitate children with autism spectrum disorders in learning distrust and deception? Retrieved November 30, 2020, from https://www.sciencedirect.com/science/article/pii/S0747563219301487