5-2021

Robot Object Detection and Locomotion Demonstration for EECS Department Tours

Bryson Howell
bhowel13@vols.utk.edu

Ethan Haworth
University of Tennessee, Knoxville, ehaworth@vols.utk.edu

Chris Mobley
University of Tennessee, Knoxville, cmobley4@vols.utk.edu

Ian Mulet
University of Tennessee, Knoxville, imulet@vols.utk.edu

Robot Object Detection and Locomotion Demonstration for EECS Department Tours

Detailed Design Report

Bryson Howell, Ethan Haworth, Chris Mobley, Ian Mulet

Table of Contents

Robot Object Detection and Locomotion Demonstration for EECS Department Tours

Electrical Engineering and Computer Science Department, University of Tennessee Knoxville

Bryson Howell, Ethan Haworth, Chris Mobley, Ian Mulet

Executive Summary

The goal of this project was to create a demonstration that entices potential EECS students to commit to enrolling at UTK. The demonstration involves a NAO robot which is programmed to give a brief introduction to prospective students and visitors then give a demonstration of its various capabilities, including object recognition, locomotion, and text to speech. We hope that this simple demonstration will present students with what kind of work they can accomplish at the UTK EECS program and encourage them to enroll.

NAO is a small humanoid robot that is present in the Distributed Intelligence Laboratory found within the EECS department. This robot stands at 23 inches tall and sports two front-facing cameras, four sonars, two lateral microphones, two speakers, and 25 degrees-of-freedom movement. The demonstration consists of the NAO robot performing a short introduction speech before moving on to an object recognition task in which it views a series of randomly placed objects on a table and then speaks the identified objects' names to the audience. After this, it will move on to a locomotion task in which the robot searches for and locates a soccer ball, walks up to it, and kicks it away. When the demonstration is complete, the robot will give another brief "goodbye" speech.

This project is built around five core engineering characteristics. Four of these are variables: accuracy of finding the ball, kicking the ball, object detection, and object identification. The last one is a constraint: the safety of the demonstration.

Problem Definition & Background

The idea of facial and object detection has been a recent ideation within the EECS community, and really has taken off in interest in the last 10 years [1]. In 2005 the dataset from INRIA became one of the most popular and cited datasets for pedestrian detection. This dataset was released in the early years of object detection. As time went on the technical evolution of object detection exploded and a more unified philosophy came to fruition. The application of object detection is found everywhere in modern day life. Whether it be for video annotations, facial recognition (Facebook for example), self-driving cars, classification of images, reverse searching of images, product manufacturing, and homeland defense [2].

While the idea of object detection has become more mainstream, it does not make applications of this technique any simpler to implement. There are numerous variables that go into object detection. For example, light levels, sensor type, object geometry, and color all play a role in if the detection task is successful or not. There are many variables that go into the simple task of identifying a small colored object. To account for the increasing number of contributing factors, there are several different methods of object detection. These are usually split into two different approaches, machine learning approaches and deep learning approaches. In the case of deep learning approaches, being able to do end-to-end object detection without predefining object features is through convolutional neural networks [3].

To efficiently do object detection with the variables stated, we would need to select which algorithm and approach should be taken. The objects that are going to be used will have defined features, mainly color and shape, but either method is viable for our constraints. If we can select a method that enables us to work synergistically with cameras and the environment that NAO is in, it would proceed smoothly.

The primary approach would be working with a deep neural network API, with a focus on object detection. This can be accomplished by utilizing the RoboDNN, which is most commonly used within Python and Pytorch. Pair this with our own communication system with RoboDNN and we can communicate to obtain object detection via learning process.

Another approach would be using the nodes that are located within the Webots software bank. If the nodes can be utilized, then we can algorithmically identify objects within one program rather than having several API calls to another program, or having to do learning and weights for import. This is the more feasible choice as it gives simplicity to the problem, rather than complicating it as several API's running at once (less change for failure).

Changelog

There are several things that have changed from the initial proposal to the finished product. In this section I will go over the design choices, problems, and decisions we made.

The first thing that we changed was the departure from using the Meka Robotics M3 mobile humanoid robot (aka. Rosie) to using a NAO. The NAO has far more documentation, is cheaper, and is easier to work with than Rosie. Rosie was a custom made robot so working with her proprietary software was quite difficult, so we decided to switch to the more approachable NAO robots. In addition, it would be simple for another group to pick up where we left off and extend the project using the NAO located in the Distributed Intelligence Lab.

The next change we decided to make was the removal of the grasping section of the demo. The reason being that the NAO has very small appendages compared to Rosie, making it difficult to pick up and manipulate objects. Instead, we decided to include a locomotion objective where the NAO locates and kicks a ball instead.

Another change that was made was the removal of a facial recognition task. This was decided after a several week struggle which started in mid February when we started with object recognition. The feat stood too large to also include facial recognition in the final product.

Voice recognition changed from recognizing the voice to having a pre recorded speech that is given before and during the demo to the students.

Many of the reasons that these changes occurred was due to the large task of handling object detection. This proved a much greater problem than we originally imagined it to be last semester. Many of the changes occurred throughout the semester as we had to axe more things to focus on object recognition. If we were to give a timeline on when the changes were made it would be the following: switching to NAO in mid January, similarly with the grasping section, facial recognition was axed in middle of February, and finally voice recognition in April.

Lastly, it was concluded due to COVID-19 preventing us from meeting in person consistently, we are going to utilize Webots for our simulation and that the project will be based on the simulation solely. If this were to be implemented in a real NAO we would have utilized choregraphe as an alternative as it gives much better functionality for connecting to NAO's and simpler node design.

Requirements Specification

The final product will be a demonstration where a NAO robot gives a short introductory speech, identifies objects and speaks their names, detects and moves towards a ball and kicks it, and then gives another brief farewell speech.

The customer requirements for the project are covered in Table 1. As of Spring 2021 there have been changes in the engineering characteristics and customer requirements, please look at the change log and the updated Table 1.

As for the customer, this is specifically for a lab at UTK and not meant for a large customer base. While that is the case, this application can be applied at several different establishments such as museums, other labs, or Universities. The actual robot itself is a significant investment, so we are treating this as a prototype since access is very simple. This is a project that is working in conjunction with the Distributed Intelligence Laboratory that houses the NAO's in the EECS department.

While the project is designed there are several variables and constraints that need to be addressed for the design specifications. The five most significant engineering characteristics are shown in Table 2. This was formed by examining the goals and requirements and determining what EC's would be needed for the final product. Table 2 has been updated to reflect the changes from the change log.

Demonstration of NAO for touring students: NAO should be able to function during each lab tour that features potential students for the EECS department.
Object Detection: NAO should be able to detect objects varying in color and shape at a range of ~5 meters.
Vocalization: NAO should give a speech at the beginning of the demo to students. NAO should ask the user for input at various sections of the demonstration.
Voice Recognition: Cannot be achieved in simulation. If given more time to complete the project this would be implemented on hardware.
Movement: The NAO should be able to walk and change directions. Furthermore, the NAO should be able to kick a ball from a variety of orientations and positions.

Table 1. Customer requirements for the Object Detection and Grasping Demonstration, listing in order of importance (top is the highest priority)

Engineering Characteristic	Units/Options	Significance
Accuracy of Object Detection	Percentage of Success	Level of success in regards to detection of an object
Accuracy of Object Identification	Percentage of Success	Level of success in regards to identifying an object
Accuracy of Finding Soccer Ball	Percentage of Success	Level of success in regard to finding and moving to a soccer ball
Accuracy of Kicking Soccer Ball	Percentage of Success	Level of success in regard to kicking the soccer ball
Safety	Damage to surroundings or robot	Constraint: Making sure that the students, environment, and robot are safe during a demonstration.

Table 2. Top 5 engineering characteristics for the project

Our first three characteristics are all related to the accuracy or success rate of various tasks in the demonstration, and are variable based on the reactions of the intended audience of the demonstration. We will need to do some market research to determine what an acceptable completion rate is, but currently we would like our demonstration to have a success rate of at least 75%. Our design constraints are based on our access to the robot and the safety of everyone present. Our demonstration will be considered a failure if there is any significant chance that it could damage the robot or cause harm to any lab visitors or EECS faculty.

Technical Approach

The original objective of this project was to create a working demonstration of different tasks common to robotics research, and evaluate the effectiveness of the demonstration at representing the research done at UT and inspiring prospective students. The primary component of this demonstration would have been some implementation of an object detection algorithm. The object detection method used was mainly dependent on the sensors available on Rosie, i.e. a depth camera. The plan for the detection task was never fully defined as the group shifted focus toward the NAO demonstration relatively early this semester, but it would have been complex enough as to be non-trivial. In addition we had planned to add a speech recognition component to our demonstration, in which the robot would have identified objects based on verbal commands from a human assistant. Additionally, once we were able to build a successful object detection feature, our plan was to add a locomotion task to the demonstration. The simplest locomotion task would have been for the robot to point at a specified object, but we could also have the robot grasp a specified object. This grasping task would change the design considerations of the object detection task, as the robot will have to visually identify the best grasping point on each object.

Each of these components were to be implemented using Python, but the programming language for each individual component can vary. We planned to build our demonstration system using the Robot Operating System (ROS) [4]. ROS is a framework for developing robotics applications, and will be used to manage the flow of data from our robot's sensors to the various ROS nodes, each of which will contain a behavior necessary for the demonstration.

Additionally, we focussed on the use of simulations to test our code rather than the physical robot due to limited access to the robot and facilities. Also, using a simulator reduced the risk of damage occurring to the robot during testing. There are several simulators compatible with ROS, such as Gazebo [5] or Webots [6]. Ideally, these simulators will allow our code to easily transfer from the virtual environment to the real world.

In the end, however, the direction of this project shifted drastically over the course of the spring semester. As described in the changelog section, we ended up shifting our focus away from using the Rosie robot towards using the smaller and more manageable NAO robots.

While the overall goal of creating a demonstration for visitors to UT's campus remained the same, the tasks involved with the demonstration changed dramatically as the NAO's smaller frame prevented the more complex tasks made possible by using Rosie. In addition, the NAO's sensor suite is not quite as complex as Rosie's and thus NAO cannot perform movements as precisely as Rosie. Finally, as meeting in person as a team proved to be challenging this semester, the decision was made to focus completely on the simulation of the demonstration as opposed to working with the physical robot directly.

After trying out a couple of simulators, Webots was selected to perform the robot and world simulation. This was chosen because it is widely used and thus it is well documented especially in regards to simulating NAO robots. In addition, it was able to effectively simulate all of the NAO's sensors and cameras as well as all its articulations.

The controller for the simulated NAO was written in Python and used the built-in Webots Robot, Motion, Speaker, Camera, and CameraRecognition packages to draw sensor data and control the NAO. In addition, we used keyboard input to prompt the user on which actions the NAO should perform.

Design Concepts, Evaluation & Selection

There are two designs that our group originally came up with when focussed on the Rosie demonstration. Both of the designs utilized the same budget (meaning the same objects will be used) the difference between the two designs is the method used to detect the objects. These methods considered were via machine learning and deep learning.

For Design 1, we planned to use a machine learning approach. The idea for machine learning is that first a framework finds the defining features of an object. Next, these defining features are passed to a support vector machine (SVM) which further classifies the objects based upon their defining features [7].

For design 2 the deep learning approach would have been used. Deep learning has the notable quality to do end-to-end object detection without needing to specify defining features and is based upon convolutional neural networks (CNN). There are numerous individual methods to implement CNN and one can be selected as further research is decided [8].

After switching focus towards using the simulated NAO robots in Webots, a third object detection and identification method became apparent. This was using Webots built in CameraObjectRecognition package. This uses an algorithm to detect pixels on the camera which then produces an estimate of the size of the object. The colors and shape are then run against known objects and the details of the object are identified. Using this method is much simpler than the other 2 designs, and it uses all in house methods. Both of the other methods would require including web services and object weights would need to be calculated. In Webots, however, all the weights and necessary information are pre-calculated and readily available.

For each of the major demonstration components described in the previous section, we must choose some design concept that will accomplish the task while also being reasonable to implement within the given timeframe. These major design decisions, and the concepts for each one, are summarized in Table 3.

Demonstration Component	Concepts	Engineering Characteristic
Object Detection	1. Object name 2. Object shape 3. Object color	Success rate of object detection
Locomotion	1. Grasping 2. Kicking motion	Success rate of movement

Table 3. Major Design Decisions

For the object detection component, we expect the highest success rate will come from identifying an object by name, as identifying an object by its traits would require the use of extra deep-learning packages and additional training outside of Webots. Additionally, the NAO is currently designed to detect and kick stationary targets. A stretch goal of ours was to detect and kick a moving target, but the dynamic movement and object detection proved to be much more complex than expected and thus was unable to be implemented.

Object Detection			
Design Concepts	**Time Cost**	**Complexity**	**Score**
Name of Object	3	2	5
Object Shape	4	4	8
Object Color	1	5	6
Locomotion			
Design Concepts	**Time Cost**	**Complexity**	**Score**
Grasping	3	5	8
Kicking	1	3	6

Table 4. Decision Matrix for Design Concepts

Out of the locomotion tasks, basic movement was relatively simple as there are several predefined movement plans for taking steps, strafing, and turning. The main struggle of the ball kicking task was determining the ball's position and determining the best path towards it. A solution for this was able to be found, but the process could be made more efficient by developing dynamic movement functions. However, this was beyond our current capabilities.

We will need to select one option for each of the demonstration components as shown in Table 3 for our final deliverable. The evaluation of each option is presented in Table 4.

In the end we decided that the best way to demonstrate the NAO was to perform multiple tasks in the demonstration. The first was to face various objects and speak their name when they are presented to it. The next was a locomotion task in which combined object detection as well as movement. This task involved the robot detecting a soccer ball, walking up to it, and kicking it.

Embodiment Design

There are several different modules that interact during the process of this demonstration, which has mainly been revised to just include object detection and movement. The first main module that will be talked about is Webots.

Webots is an integral part of testing when using robotics as it allows all the other programs (which will be discussed later) to interact and simulate the design that is being proposed. Firstly, the inclusion of the robot design in Webots is built into the system, meaning that the NAO already has a model that is found within Webots. Webots then lets you make a scene, including objects, floors, walls, and robots. We made a simple scene that contains the NAO and the objects. The next step was to create a controller for the NAO, which would prove to be difficult. The controller would include the access to the sensors, actuators, and camera. This would enable the robot to move, and actively display footage to the controller in real time. Luckily, Webots has a library which makes some of the coding easier. The controller was created, enabling use of the cameras, accelerometer, gyro system, GPS, interatial system, LED's, and actuators (which move the robot). After this is set up the robot is accurately simulated in a scene, the movements and sensors are calibrated as they are in real life. Some of the inputs that Webots takes are motion files, object files, robot models, and NODES. One thing is, while motion can be handled by Webots, that is not necessarily the only option, and we opted for another route. Although I think it is important to recognize the benefit of using Webots for motion, as in house everything is simulated in the program, and the method that we used has deprecated and is supported by users but not the company any longer. That leads me to the next module: choregraphe.

Choregraphe [10] is made by the same manufacturer as the NAO robot, and is primarily used to link the NAO to the computer to add behaviors. It is an essential process to connect to the NAO, and is a main way to create nodes that add behaviors to the NAO. When you create nodes they take input from the actual machine and provide outputs into other nodes to link a sequence of events that control the NAO. Now that is not necessarily the only use for Choregraphe, as you can also create behaviors that utilize object detection. Using the nodes, and external python code to dictate the nodes it is possible to make object detection a behavior for the NAO. It takes behaviors by processing a .xar file. It outputs an IP address that communicates with the NAO robot. Along with the object detection, it also dictates the motion of the NAO based on other variables and has a wide array of options that are yet to be explored by the group. Although, the kicking motion and walking motion shall be handled in choregraphe. Partially, we are unsure if we should handle all the object detection and movement in choreograph, or leave the motion in webots.

The method mentioned above about how we are going to create the connection between movement and object detection is by using choregraphe and webots. Luckily we can use webots as a simulator for choregraphe by using a deprecated utility called NAOqisim [9], which is based off of NAOqi. NAOqi is included in the base choregraphe program and is what enables the easy access to motion and movement. NAOqisim allows that to be simulated in Webots instead of having to connect to a real robot, in our case we opted for this method since Covid-19 has hindered our ability to visit labs in Min-Kao. NAOqisim operates very simply by creating an virtual IP address in Webots and connects it to choregraphe allowing it to function just like a real robot. In that way we can directly test while being in a group call, easily debugging and keeping the scene exactly the same. It is important that the variables that change can be reverted, as we want to keep everything constant. Webots allows you to revert the scene after movement of an object occurs bringing great benefit to debugging the object detection and movement. Likewise we can also set up various different cases for the NAO, although in our demo it is a very simple demonstration as most of our time was spent getting everything running. The process has been laid out so iterating on it would be simple.

Figure 1 shows the flowchart of our robot demo. Walking through the demo, the NAO first gives a welcome speech. In the actual program the speech function simply takes in a string and emits it via text-to-speech. The NAO then moves on to the recognition task. NAO is transferred between tasks using the Webots Supervisor API, but in a hardware demonstration a human user would have to physically move NAO between locations. It will identify the objects that are randomly placed on the table. In the actual code, the recognized objects are loaded in as nodes, each of the nodes is a struct which contains information of the object like: ID, position, model, colors, size … etc. We obtain the model string and use that as a basis for the NAO to say what object it is currently detecting. The user then prompts the NAO to either move forward with the demo or play the recognition task again.

The NAO then moves on in the demo to the soccer kicking task. In the program we teleport the NAO to the 'arena' where that occurs, but if this were to be translated to real time we would simply have the NAO moved, and a soccer ball placed somewhere in front of it. Moving on, the NAO will look for the soccer ball, and if the ball is not found the NAO will rotate and repeat until a ball is found. If it does find a ball the NAO will walk towards the ball until standing in front of the ball. The NAO then kicks the soccer ball, and the user is prompted to either repeat that section or finish up the demo by giving a goodbye speech. During the kicking portion of the demo, the NAO simply tries to recognize the soccer ball. Once the ball is in its vision the NAO will obtain relative coordinates and walk to the ball, until the ball disappears from view, signaling the ball is in front of the NAO's feet. The kick motion is then signaled to the NAO and the ball is kicked.

In terms of why the NAO was selected instead of our previous decision of using Rosie was a couple of reasons. First, documentation and compatibility with several programs that are available are nonexistent. Second, is that NAO is a much simpler robot in terms of size, control, and price. If we were to damage Rosie, getting a replacement would be nearly impossible since the company that made her has gone out of business, where in comparison NAO's are still being made and have plenty of documentation. Given the choice we decided to utilize the NAO instead of Rosie, enabling us to use choregraphe and webots along with plenty of documentation for each. Another reason is the method of connection, as having an IP that the NAO emits enables easy connection to test the robot. While we were not able to interact directly with the NAO, once Covid-19 restrictions are lifted it would be quite trivial to access the NAO with authorization.

Test Plan

For recollection the five engineering characteristics that we selected were the following.

Four variables: finding the ball, kicking the ball, object detection, and object identification.

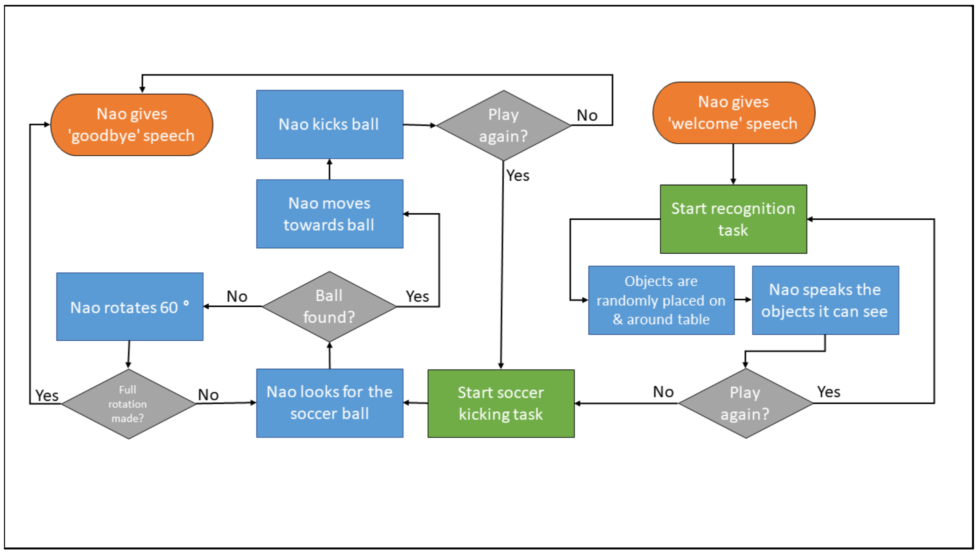

Nao gives 'goodbye' speech
Nao kicks ball
Play again?
No
Yes
Nao gives 'welcome' speech
Nao moves towards ball
Start recognition task
Nao rotates 60 °
No
Ball found?
Yes
Objects are randomly placed on & around table
Nao speaks the objects it can see
Yes
Full rotation made?
No
Nao looks for the soccer ball
Start soccer kicking task
No
Play again?
Yes

Figure 1. A flowchart of the NAO's demonstration plan

One constraint: safety of the demo.

The first test revolves around the movement and kicking motion, and specifically kicking the soccer ball placed in the world. Our plans for testing the first two variables are rather simple in design, but will give us the data we need to judge its performance appropriately. The plan is to have a soccer ball present, identify it to get its relative distance, walk towards it, perform a kicking motion, and record if the kick was successful. A kick is designated as successful if the ball's speed is greater than zero after the kicking motion is completed. We return a boolean variable if the ball's velocity changes from 0 which is recorded as our experimental result. Likewise, for the ball identification we return a boolean variable that represents if the soccer ball was detected. It is important to note that if the ball is never detected the NAO will never seek to move towards the soccer ball and kick it. If this case occurs, finding a ball and kicking it will be recorded as a failure.

The second test revolves around testing the vision, specifically object detection and object identification. The world will be set up so that there will be four different tables each scattered with objects of different size, shape, and color. The NAO will stand before the objects and try to detect and identify the objects. Similarly, if an object is detected a boolean will be returned that an object is detected. Furthermore, if an object is detected the object name will be present in the camera overlay and the ID of the object will be returned, so we can accurately determine if the object was identified. If the object name is spoken by NAO that is a success, likewise if it is not that is a failure. If the object is never detected that will also be recorded as a failure for object identification since the two work in tandem.

The last EC is the safety of the demo. This is being classified with several factors. First is human safety, and while this is an important consideration the NAO's should not present a large risk towards humans. The second most important reason is protecting the safety of the NAO. NAO's by nature are rather clumsy as the motions need to be perfect to keep them upright, and even then disturbances could knock one over. The NAO is expected to take a light blow and still work perfectly, but to keep it on the safe side preventing the NAO from falling over should be a priority. We will record how many times the NAO falls over during the simulation, specifically during the kicking segment. We will record this information while performing the soccer kick tests.

Please reference the appendix tables 8 and 9 for the results of the testing.

Deliverables

This project produced a working prototype that is able to handle a basic NAO robot demonstration once it is ported onto a physical robot. Furthermore, the following deliverables are available from the demonstration:

- A robot able to give introductory and farewell speeches to visitors
- A robot able to detect and identify a soccer ball, move to it, and kick it away
- A robot with the ability to detect objects, and identify them.
- Functioning code/software that manipulates the robot into performing the said tasks above.
- Statistics noting the accuracy of the stated variables above, in the format of a report.
- A final report on the performance of the demonstration, and detailing the successes and failures of the concurrent project.

For the current changes to the deliverables, please look at the changelog in the beginning of the paper. Many of the deliverables have changed but the core idea of a robot able to perform object detection and giving a demo speech is the updated deliverables.

Project Management

A tentative schedule has been planned to assist in the completion of the project according to the requirements and deliverables stated above. The schedule itself does not include the fact that spring break, finals, and other projects will affect the timeline stated below. Furthermore, the goal for the rest of the current semester is the creation of the presentation. The table follows a weekly schedule with an individual goal slated for each week.

Since everyone in this group is a Computer Science major, we shall work together collaboratively on each goal for each week. We think that since each task is large enough, splitting them down further and assigning individual tasks would greatly improve speed, and produce more clean and efficient code. With the presence of each member working on the same section enables our group to more efficiently manage bugs and details that are unknown. Hopefully this method will enable our group to learn at a much faster pace since ideas will be bounced off each other.

Week	Goal
1	Research Object Detection Methods
2	Research Implementations
3	Research Implementations
4 (Note: this is when we switched robots)	Research NAO
5	Research NAO
6	ROS Setup/Testing
7	Webots Setup/Testing
8	Object Detection
9	Gazebo Setup/Testing
10	Object Detection
11	Locomotion
12	Locomotion
13	Speech
14	Final Presentation

Table 5. Week by week schedule for Spring 2021 based upon each goal.

Much of the original timeline was delegated to voice recognition and grasping, but stated above in the change log those features were axed. The new schedule reflects the actual goals during the semester. Object detection took much longer than we had anticipated, and took up 95% of the time.

Budget

The following expenses are estimated:

Item:	
Shapes (wooden, by color and shape)	25.00
Household Items	35.00
Speaker	25.00
NAO	7990.00
Total	8075.00

This is just an estimation on what objects *could* be bought, as some of these can be procured for free. The addition of the NAO robot is new, as this is the estimated cost it would take for an entity to model a design similar to ours. Luckily for us the NAO was simulated, and even if it was not the DI lab has NAO's that we could have used.

References

[1] Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., & Qu, R. "Object Detection in 20 Years: A Survey" *IEEE Access 7,* 2019

[2] Ling Guan; Yifeng He; Sun-Yuan Kung *Multimedia Image and Video Processing*. CRC Press. pp. 331, 2012

[3] Ross, Girshick. "Rich feature hierarchies for accurate object detection and semantic segmentation". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE: 580–587, 2014

[4] " Robot operating system", ros.org, 2020 [Online].

[5] "Gazebo" gazebosim.org, 2020, [Online]

[6] “webots” cyberbotics.com, 2020, [Online]

[7] Cortes, C, Vapnik, V. “Support-vector networks” *Machine Learning*, 1995

[8] Valueva, M, Nagornov, N, Lyakhov, P, Valuev, G, Chervyakov, N, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation". *Mathematics and Computers in Simulation*, 2020

[9]“Naoqisim”,https://github.com/cyberbotics/naoqisim, 2021, [Online]

[10]”Choregraphe”,http://doc.aldebaran.com/1-14/software/choregraphe/choregraphe_overview.html , 2021 [Online]

Appendix 1 - Definitions

Support vector machine (SVN) - supervised learning models with learning algorithms that analyze data used for classification

Convolutional neural network (CNN) - a class of deep neural networks, most commonly applied to analyzing visual imagery

RoboDNN - a fast, forward-only deep neural network library designed for Nao robots

Rosie - a Meka Robotics M3 mobile humanoid robot found in the Distributed intelligence laboratory found in the EECS department

Robot operating system (ROS) - the system that controls the robot

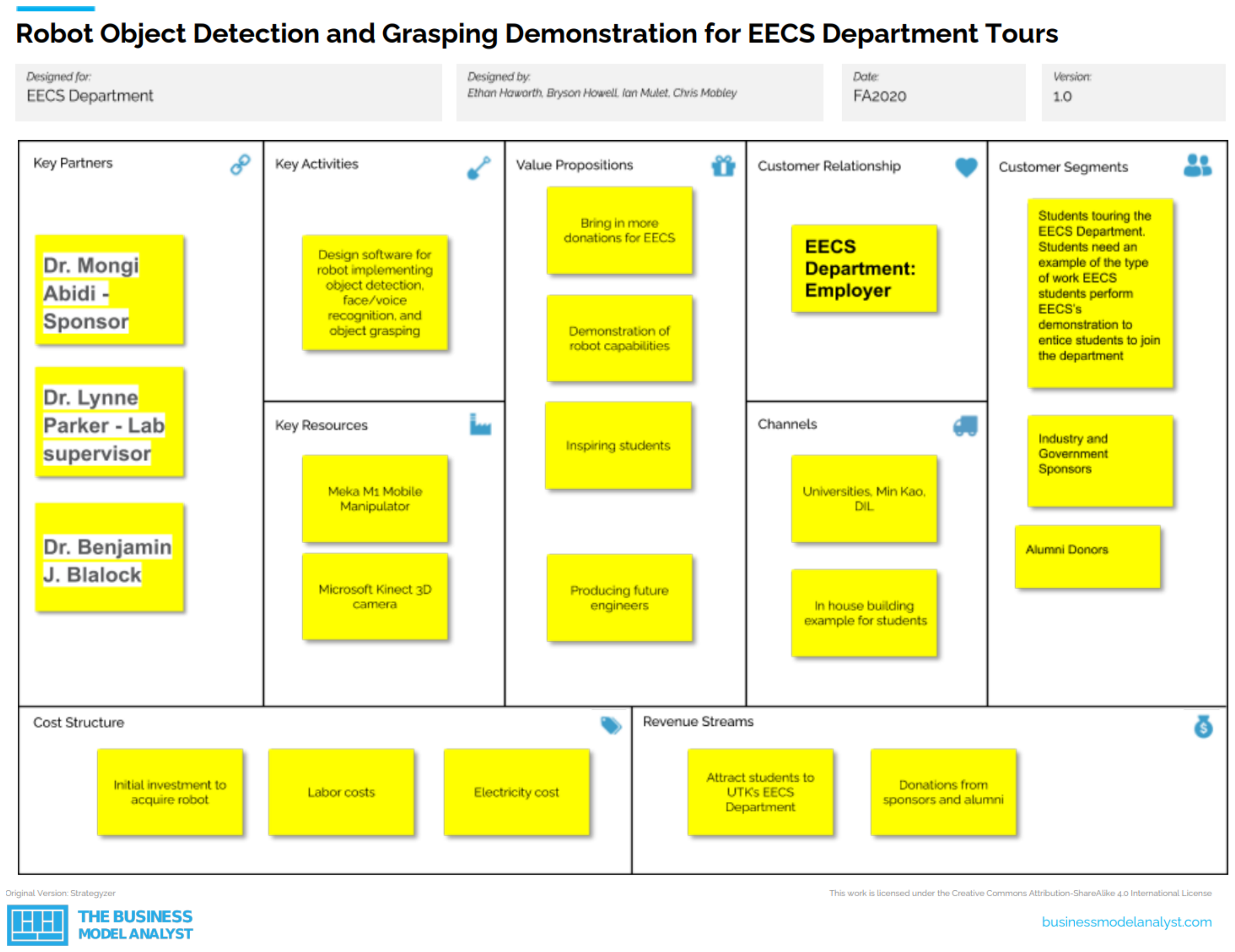
Gazebo - robot simulation software able to design and test algorithms for robotics

Webots - open source software to simulate robots movement

Nodes - objects and behaviors used within the Webots software to simulate various effects

NAO - a robot in the Distributed intelligence lab found in the EECS department. It is developed by Softbank and is a version 5 model.

Appendix 2 - Business Model Canvas



Robot Object Detection and Grasping Demonstration for EECS Department Tours

Designed for:	Designed by:	Date:	Version:
EECS Department	Ethan Haworth, Bryson Howell, Ian Mulet, Chris Mobley	FA2020	1.0

Key Partners
- Dr. Mongi Abidi - Sponsor
- Dr. Lynne Parker - Lab supervisor
- Dr. Benjamin J. Blalock

Key Activities
- Design software for robot implementing object detection, face/voice recognition, and object grasping

Key Resources
- Meka M1 Mobile Manipulator
- Microsoft Kinect 3D camera

Value Propositions
- Bring in more donations for EECS
- Demonstration of robot capabilities
- Inspiring students
- Producing future engineers

Customer Relationship
- EECS Department: Employer

Channels
- Universities, Min Kao, DIL
- In house building example for students

Customer Segments
- Students touring the EECS Department. Students need an example of the type of work EECS students perform EECS's demonstration to entice students to join the department
- Industry and Government Sponsors
- Alumni Donors

Cost Structure
- Initial investment to acquire robot
- Labor costs
- Electricity cost

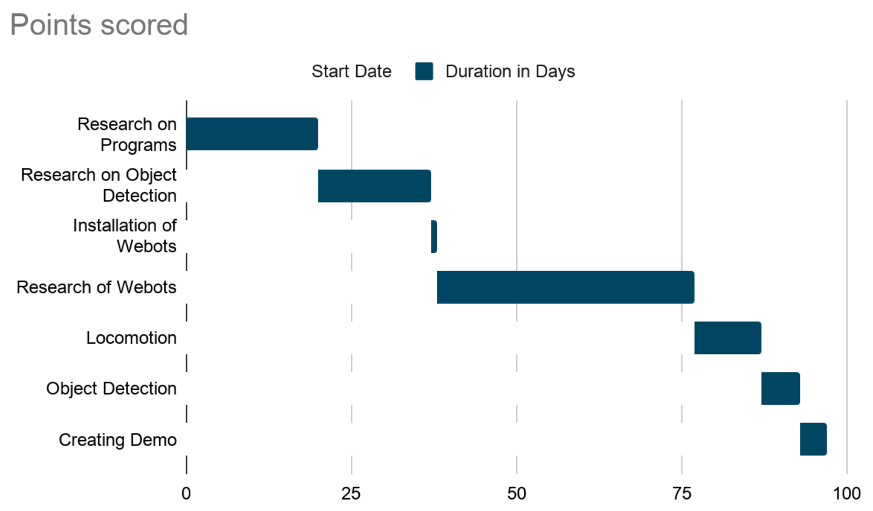
Revenue Streams
- Attract students to UTK's EECS Department
- Donations from sponsors and alumni

Original Version: Strategyzer

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License

THE BUSINESS MODEL ANALYST

businessmodelanalyst.com



Test Plan Matrix

Test Plan	Objective	Milestone
Detecting the soccer ball	Make sure that the soccer ball is properly detected by the NAO robot	Marks beginning of the detection phase, furthermore, makes use of the depth functionality
Kicking the soccer ball	Make the NAO robot kick the soccer ball	Enables the final testing process for the locomotion to occur. Letting us test for bugs in our locomotion
Detecting objects	Make the NAO detect objects on the table	Partially covered by the first test plan, but a wide range of objects should be detected.
Identifying objects	Make the NAO be able to identify the objects that are detected.	Clears the final goal to tie the entire demo together.

Table 6. Test Plan Matrix

Quad Chart

Objective: Design a demo testing the functionality and demonstrative properties of the NAO. Main object being to entice students with a fun demo and show the work of an EECS student. The Goal of testing is to be above 65% success rate with a +- 5% margin of error, due to the nature of the NAO. The main demo will be with the NAO robot.	**Key Customers:** The key customers are the EECS students that are touring the department. The key partner is the EECS department lab, Distributed Intelligence Lab This could be distributed to several customers that desire a personalized demo for any occasion.
Approach: Using webots, a simulation of the demo will be created. This is mainly using a controller and nodes that manipulate the NAO. It utilizes object recognition in the camera overlay, speakers, movement, objects, and the robot.	**Key Milestones:** Requirements, research, design 3/5/2021 Locomotion 4/13/2021 Object Detection 4/23/2021 Demo completion 5/3/2021

Table 7. Quad Chart

Soccer Ball Tests

	Ball Found	Ball Kicked
0	TRUE	TRUE
1	TRUE	FALSE
2	FALSE	FALSE
3	TRUE	TRUE
4	TRUE	TRUE
5	TRUE	FALSE
6	FALSE	FALSE
7	TRUE	TRUE
8	FALSE	FALSE
9	TRUE	FALSE
10	TRUE	FALSE
11	TRUE	FALSE
12	TRUE	TRUE
13	TRUE	TRUE
14	TRUE	FALSE
15	TRUE	FALSE
16	TRUE	FALSE
17	TRUE	TRUE
18	TRUE	TRUE
19	TRUE	TRUE

Table 8

Object Recognition Tests

	% Seen (Scene 1)	% Named (Scene 1)	% Seen (Scene 2)	% Named (Scene 2)	% Seen (Scene 3)	% Named (Scene 3)	% Seen (Table 4)	% Named (Table 4)
0	1	1	1	1	1	0.8	0.6	0.6
1	1	1	1	1	0.8	0.6	0.6	0.6
2	1	1	1	1	0.8	0.6	0.6	0.6
3	1	1	1	1	0.8	0.6	0.6	0.6
4	1	1	1	1	0.8	0.6	0.6	0.6
5	1	1	1	1	1	0.8	0.6	0.6
6	1	1	1	1	0.8	0.6	0.6	0.6
7	1	1	1	1	0.8	0.6	0.6	0.6
8	1	1	1	1	0.8	0.6	0.6	0.6
9	1	1	1	1	0.8	0.6	0.6	0.6

Table 9

[illegible]

[illegible]

Chris Mobley

Ethan Haworth