ROBOTICS CURRICULUM PROJECT

August 04, 2009
Rev 1

Final Report

Steven Elia | Siying Diana Hu | Sixiao Joy Liu

Technology Consulting in the Global Community

Our Lady of Lourdes College Foundation

Carnegie Mellon University

Table of Contents

EXECUTIVE SUMMARY

Background Information

Our Lady of Lourdes College Foundation (OLLCF) is an ever-expanding educational institution with the goal of furthering the development of capabilities for the Filipino youth and community. Started in 1977 as a nursing school, today it has expanded into nine colleges along with primary and secondary education institutes composed of 112 faculty members led by the founder and his five children serving as the board of trustees. Fueled by their passion for technology, OLLCF looks forward to introducing a robotics curriculum at the high school and college levels incorporating a hands-on approach towards learning. It hopes to inspire the student body—composed of 150 high school, 40 elementary school, and 2000 college students, of which 130 are in the College of Computer Studies—to strive toward technical innovation in order to advance the Filipino community.

Consulting Tasks

OLLCF hopes to offer a robotics curriculum in order to further the college's goal of becoming an innovative educational institution in the Philippines and fulfill its mission of nurturing leaders who will advance the country technologically. OLLCF has asked us to include robotics topics in an introductory course for its high school and college as it currently does not have the capabilities to develop such programs itself.

In order to accomplish this, we first evaluated the current capabilities of the college and high school. This included understanding OLLCF's educational system, seeing where the classes could fit into its existing curriculum, understanding how classes are taught, knowing what faculty members are available, understanding the teachers' and students' capabilities and foundation, knowing what facilities and equipments are available, and knowing the administration's desires and vision for the school.

Since OLLCF is very eager to incorporate hands-on learning into the teaching curriculum, we analyzed different alternative tools available on the market that would best fit the college's needs. It is worthy to point out that the best tool on the market might not be the best tool that fits OLLCF's needs. Therefore, it was absolutely essential to understand the capabilities and needs of OLLCF and work with the faculty and administration to determine the hardware to purchase for the curriculum depending on topics that could be taught with it and its ease of use, support, and price.

Once we agreed with OLLCF on what set of robotics topics the curriculum should cover and what hardware will aid in that instruction, we planned to hold a series of daily workshops for selected faculty members intended to give the expertise necessary for teaching introductory robotics classes. Further detailed instruction that they seek for their own benefit was included as much as possible.

Alongside training the personnel involved, our workshops also offered the opportunity to collectively discuss methods for teaching robotics material, including demonstrations and lab exercises. This group environment further allowed those parties involved to share any expected difficulties at each level of instruction and plan for questions that students taking these introductory classes might ask. Such activities enabled us to finalize the curriculum for an introductory class at both the high school and college levels in tandem with the faculty members and help those who will eventually teach the robotics courses develop their own lesson plans on the subject matter. This guarantees that each instructor is able to prepare a class which is consistent with his or her teaching style, thus ensuring the courses can be taught in a sustainable manner for future years.

We initially evaluated the current equipment in the maritime engineering and computer science departments to determine what OLLCF already has that can be included in a robotics laboratory. After completing the high school and college level curricula, we recommended a complete setup for the lab with other necessary equipment and materials that will enable students to gain hands-on experience in conjunction with the proposed curricula.

Outcomes Analysis and Recommendations

Evaluation of Current Capabilities

RESULTS

- At the high school level, the class load was already filled and time needed to be freed up for the introduction of the robotics class. It was concluded that the current computer science curriculum can be compressed into three years and the robotics class can be taught over four quarters in the last year of high school.
- At the college level, the students' requirements are very demanding; not much room can be made. Therefore, the introductory robotics class will replace the current Modeling and Simulation course offered that uses Robo-box 3.0 as a kit.

RISKS

- Shortage of qualified staff members with an interest in robotics and quick turnover of these staff members is a major risk to the sustainability of the robotics class. Because very few colleges and high schools in the Philippines offer robotics classes currently, many of the staff members have not had previous exposure or formal disciplined training. Therefore, in order for this class to be taught, staff members need to have additional training.
- Because the primary and secondary education system encompasses only ten years, the students do not have time to develop a strong mathematical foundation before attending college and learning more advance material. This limits the depth of material that can be taught. Remedial math class can be offered the summer before students enter college to partially address the problem at the college level.

Alternative Analysis

RESULTS

- After analyzing five educational robotics sets, we concluded that LEGO Mindstorms NXT best fits the needs of OLLCF. It is widely used in educational institutes and there is a variety of freely available resources and lesson plans available on the Internet. In addition, there exists a wide range of exercises with difficulty levels varying from basic to intermediate. Therefore, the kit can be used to not only teach the introductory class, but also more advanced robotics control courses using C programming as the capabilities of OLLCF increase with time.

RISKS

- The LEGO Minstorms NXT set looks very simple, thus there might be an urge to rush to build a robot by piecing together different materials before basic and intermediate knowledge is learned.
- There are only four sensor input ports so the functionality of the robot is limited. Nevertheless, advance robot programming and theory can be implemented and taught using the LEGO Minstorms NXT. These theories can be easily applied later to incorporate more capabilities when building a robot from different materials as the competency of the staff and students increases.

Training of Personnel

RESULTS

- Five faculty members attended daily workshops held for four weeks. The topics covered included basic mechanics, basic electronic components, sensors and actuators, programming using the LEGO Mindstorms NXT interface, troubleshooting, and design processes. In addition, attendees performed exercises with different LEGO Mindstorms NXT robots.

RISKS

- The workshop was taught at an accelerated pace, fitting into four weeks the material that will be taught over a semester at the college level and a year at the high school level. The amount of knowledge retained is therefore a concern.
- Because the workshop was a side activity for the faculty members involved, they did not always have the necessary free time. Therefore, only one or two members attended consistently.

Curriculum Development

RESULTS

- Two similar introductory robotics syllabuses were created. The one at the high school level consists of the same material as that taught at the college level. However, it contains more robot lab exercises and the material is taught at a slower pace. These two courses aims to provide a brief understanding of what robotics is and develop fundamental elementary robotics skills.

RISKS

- Faculty members were properly trained with an understanding of the theoretical knowledge and the programming interface LEGO Mindstorms provides and have performed those exercises available in teaching the course.

Laboratory Setup

RESULTS

- We concluded that the LEGO Mindstorms NXT sets are sufficient enough to teach an introductory course in robotics. Even if a laboratory is fully equipped, many of the tools would be left unused and scarce resources would be wasted. Therefore, we proposed a number of phases to follow in order to build up the capabilities at OLLCF so that eventually a fully equipped laboratory would be useful in the creation of innovative robots for research and practical use. It is highly recommended that these stages are followed and that the administration waits on the purchase of materials according to their usefulness as equipment tends to decrease in price as time passes and upgrades are made available. It is worthy to note that the first few phases seem to be the easiest to accomplish. However, they might be the most important in establishing a strong foundation. As the phases proceed, there will be increasing difficulty in securing the right resources, learning the material, and keeping persistence and motivation in order to become a leading robotics research institute.

- At this particular time, we recommend that 10 LEGO Mindstorm NXT sets be purchased (a minimum of 6 are necessary for OLLCF's class sizes). Large tables and a minimal of 12 computers are also needed in the classroom in order to provide ample workspace to build and test the robots that will constructed in different exercises.

RISKS

- A rush in the construction of a laboratory to maintain momentum before capabilities are built would result in wasted expenses and outdated equipment by the time the materials are finally used.

ORGANIZATIONAL ASSESSMENT

Overview

Our Lady of Lourdes College Foundation (OLLCF) is an ever-expanding educational institution with the goal of furthering the development of capabilities for the Filipino youth and community.

MISSION: "Our Lady of Lourdes College Foundation, as an institution of higher learning is dedicated to the pursuit of Knowledge, Truth and Wisdom through the Guidance of Divine Providence as it seeks the total development and formation of the Filipino Youth."

VISION: "Empowering its students with human, conceptual and technical skills so that they best develop themselves and be instruments in bringing about a GOOD and FRUITFUL LIFE FOR ALL."

It was founded by a passionate visionary patriot, Dr. Abundio P. Palencia, Sr., to develop the capabilities of the Filipino community in order to become competitive in the global arena. Today, the mission is carried out by a community led by the founder's equally passionate children serving as the board of trustees. The college seeks to concentrate on the development of technological capabilities in computing, communication, agriculture, and health care.

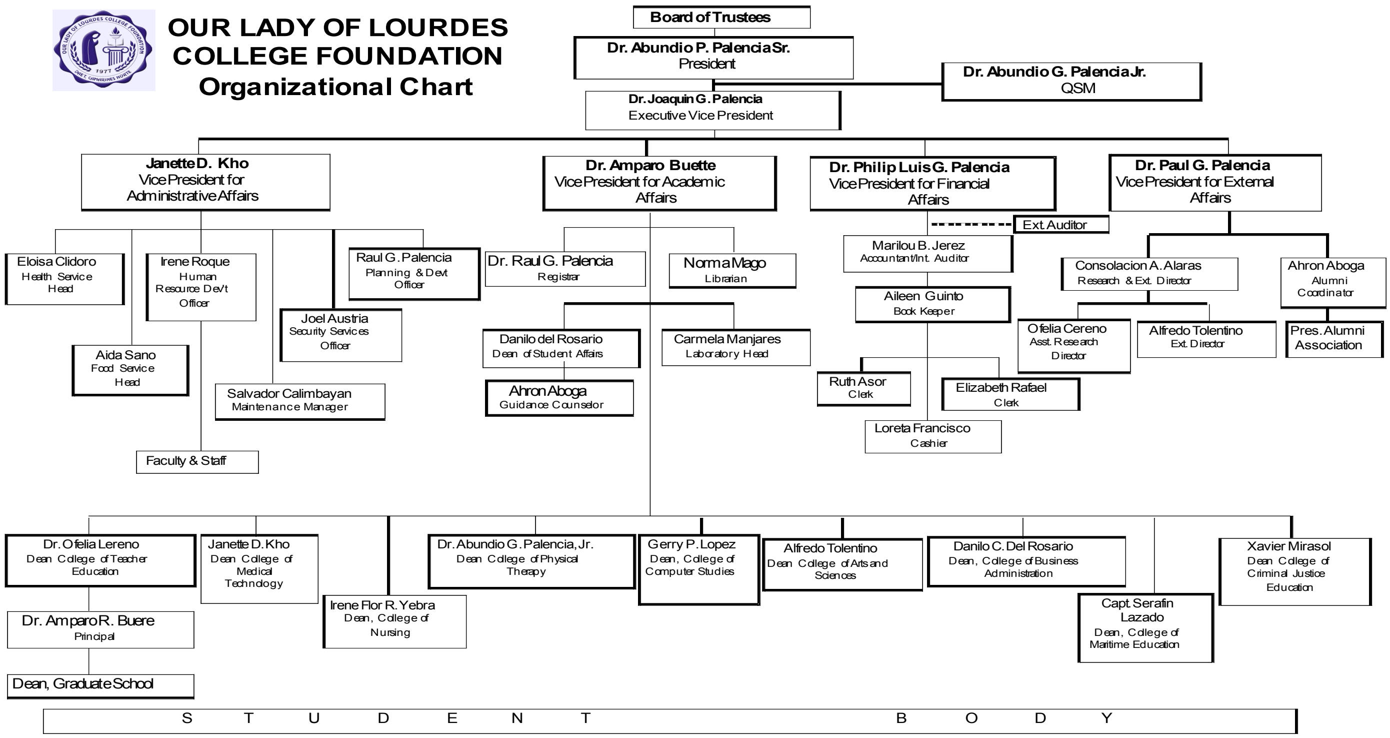


FIGURE 1: ORGANIZATION CHART

Presently, there is a basic education school teaching children from preschool throughout high school, as well as nine colleges. Three additional colleges are expected to begin in 2010.

Basic Education

Under the basic education school, there are the preschool (2 faculty members), grade school (5 faculty members) and high school (12 faculty members) departments. The high school has a particular emphasis on general sciences and computer skills.

College

OLLCF has nine colleges, each having its own structure consisting of a dean, administrative staff, and a teaching faculty. The colleges and degrees they offer are:

- College of Arts and Sciences (14 faculty members)
  - Bachelor of Science in Psychology
  - Bachelor of Arts major in Psychology and Economics
  - Associate in Communication
- College of Applied Medical Sciences (4 faculty members)
  - Bachelor of Science in Physical Therapy (5 years)
  - Bachelor of Science in Radiologic Technology
- College of Computer Studies (7 faculty members)

  - Bachelor of Science in Computer Science
  - Associate in Animation
  - Certificate in Computer Secretarial
  - Certificate in Computer Technician
  - Certificate in Computer Programming
- College of Criminal Justice Education (8 faculty members)
  - Bachelor of Science in Criminology
- College of Education (6 faculty members)
  - Bachelor of Science in Secondary Education
  - Bachelor of Science in Elementary Education
- College of Business Administration (7 faculty members)
  - Bachelor of Science in Office Administration
- College of Maritime Studies (5 faculty members)
  - Bachelor of Science in Marine Engineering
  - Bachelor of Science in Marine Transportation
  - Certificate in Welding Technology
- College of Nursing (40 faculty members)
  - Bachelor of Science in Nursing
  - Certificate in Midwifery
  - Certificate in Health Care Services
- College of Medical Technology (2 faculty members)
  - Bachelor of Science in Medical Technology

By 2010, OLLCF plans to start the College of Engineering, offering Bachelor of Science degrees for Chemical Engineering, Computer Engineering, and Electronics and Communication Engineering. A Bachelor of Science in Mass Communication under the College of Communications will also be introduced along with a Bachelor of Science in Hotel and Restaurant Management under the College of Tourism and Hospitality Management.

The college is also undergoing accreditation to become a university, which requires three levels of standardization based on Commission of Higher Education (CHED) regulations.[1] Currently, there are four programs accredited at level one: Business Administration, Elementary Education, Secondary Education, and Nursing. OLLCF is in the process of advancing from level one up to level three by means of increasing its current capabilities and incorporating a strong research component. The high school is also completing the requirements to reach level one. After three colleges reach level three, the institution can apply to become a university. Once a university has at least five colleges that are

[1] http://www.ched.gov.ph/

Centers of Excellence, it can apply to become an autonomous entity able to develop its own requirements independent of CHED. Once the university becomes autonomous, its high school can similarly gain that status after reaching level three, freeing it from the requirements of the Department of Education.

In addition, there exist partnerships with the Philippine Women's University and the University of the Philippines Open University for graduate, doctoral, and additional certificate programs.

Academic overview

OLLCF offers a complete education from preschool up to the college level. As discussed by the executive vice president, one of the school's aims is to create a curriculum that focuses on technology to provide students a competitive advantage against their peers.

High School

All students have four weekly hours of basic computer science instruction. They begin with a thorough understanding of the OpenOffice.org office suite and progress to imaging, animation, and 3D modeling with GIMP and Blender. Students are also introduced to programming, first through the C language, followed by Java and HTML. Students are able to obtain certificates in these areas once they meet the required standards (e.g., a student can obtain a certificate in C programming once he or she has achieved the requisite number of hours in related study).

College of Computer Studies

The OLLCF four-year Bachelor of Science in Computer Science (BSCS) program covers many of the same topics as comparable universities in the United States, including object-oriented programming, software engineering paradigms, algorithm design, data structures, automata and language theory, operating systems, networking, databases, and digital circuits. With the exception of programming courses, classes are taught from theoretical aspects without any labs or other hands-on experience. Moreover, as most entering students come from other high schools with little to no computer experience, many of the first few semesters are spent re-teaching topics that OLLCF covers in its high school curriculum. There is currently no procedure for OLLCF high school graduates to test out of or skip this redundant material.

Alongside many others programs, OLLCF offers the BSCS with the concept of "ladderization." This idea, popular among universities in the Philippines, is intended to accommodate those students with financial difficulties. It is common for a student to be unable to finish a four-year degree; instead of losing all of the resources put into an education upon dropping out, colleges will often offer a "ladder" to a student's intended career path. Once students reach their second year, they are eligible to take an examination from the government's Technical Education and Skills Development Authority (TESDA) to obtain a vocational certificate—rather than a bachelor's degree—in their field of study. This concept should be integrated into new curricula whenever possible because of its attraction to a greater number of students.

Technical Background

Technical Environment

Facilities include administration and faculty rooms, a student lounge, computer laboratory rooms, audio-visual rooms, a social hall, a chemistry laboratory, a physics laboratory, a microbiology and anatomy laboratory, simulator rooms, an engineering laboratory, a library, a home economics laboratory, and a play room for children.

Relevant Infrastructure

- Computers access: The school has at least one computer per administrative office and three computer labs. The labs, which are the main access point for students, have enough computers for every one or two individuals, depending on the size of a given computer class. The first lab, located in the second floor of the college and commonly called the Internet Room, is mainly used as a location with free Internet access for students and staff. It encompasses 18 computers with 2 of them dedicated for the faculty. All the computers have the same specifications: Intel Pentium 4 processor with 256 MB RAM. The second lab, used for teaching college computer science classes, is located in the second floor in the high school. This room currently houses 28 computers and is also used for teaching computer repair and basic circuit design. The third lab, located right next to the second, is used for teaching at the high school level. This lab possesses 12 computers. Computers in the second and third labs run Ubuntu.
- Network: The school has an Internet connection speed of 512 kbps. There are four installed routers, though only two are properly working wireless access points. One is located in the Internet Room in the college, and the other one is located in the president's office. The third access point is installed in the faculty office in the high school, but it currently does not work.

The fourth router is used as a switch in the computer lab located on the second floor of the high school. In each computer lab, there is a switch to provide connections for every computer. The school states that it has a contract with two different Internet providers—one for a wired connection and another for a wireless one. The ISP for the wired connection is Bayantel while the wireless one is Smart Bro.

- Future robotics laboratory: The room intended for use is still under construction. There are plans of installing a fan, and as of now there is only one electric outlet for a room that measures approximately 10 meters by 6 meters.
- Electrical power: Power outages occur sporadically, sometimes once or twice a week. They typically last five to ten minutes before power is restored or a generator is turned on.
- Local suppliers: Within Daet, there are no existing electronics suppliers. All the major ones are located in Manila. E-Gizmo Mechatronics and Robotics Central is the main supplier for the school at present (it has purchased two basic educational robotic kits there). E-Gizmo possesses all of the basic components for building robots, but it does not offer much of a variety.

Relevant Equipment

- Current robotics kit: OLLCF possesses two Robo-Box 3.0 educational robotics kits from Inex. These are currently used in the only available robotics classes in college and high school. Purchased at E-Gizmo, they are valued around $175 apiece. Each one includes a general processing board, motors, sensors, and basic mechanical parts such as gears, nuts, and bolts. The processing board interfaces with custom-made and specially fitting sensors that include range finders, infrared sensors, switches (in the form of bumpers), and light sensors. It has for 4 input/output ports, 4 sensor ports, and 8 servo motor ports. The programming language it supports is Logo, a simple graphical language that uses blocks and flow charts.
- Unused robotics kit: OLLCF owns one iRobot Roomba and a Gumstix microcontroller which have never been used. This kit was sent from the United States with expectations to include it in the future robotics curriculum. Currently, no qualified personnel in the school know how to operate or develop this robot. The knowledge necessary for this includes an understanding of Gumstix and Linux development, command modules, and advanced electronics, as well as programming in C and C++.
- Electronics tools: OLLCF owns a few soldering irons and bread boards that are unused as of now. There are prospects to utilize them in the future for the computer engineering and electronics and communications engineering programs. In addition, the maritime department has a limited number of measuring devices such as digital multimeters.

See our alternatives analysis for further details.

Technical Support

The maintenance team consisting of a manager and two technicians is in charge of maintaining all of the computer- and network-related tasks at the school. This semester, the technicians are also responsible for teaching classes due to a shortage of computer science faculty. Both Jayson Defeo and Romel Patorito are TESDA-certified technicians in second level Computer Hardware. The manager for maintenance, Salvador Calimbayan, sustains all of the physical spaces, furniture, equipment, and computers. He is a graduate in electronics and communications engineering and currently holds this position until the school opens its engineering college, at which point he is expected to be hired as a faculty member there. He has experience teaching a microcontrollers course at another university.

Parties Involved

Administration

The administration has a grand vision for OLLCF to develop students with skill sets that would enable them to be competitive in the global technological community. By doing this, OLLCF will be helping the local region build a technology-based infrastructure to increase the standard of living. The college is already pursuing this with its strong computer animation program and is looking to expand its technological focuses into other areas as well. With the College of Engineering expected to open within a year's time, incorporating a robotics curriculum would take advantage of the existing computer science program while further promoting those engineering disciplines that are directly involved. Such a robotics program would also further the administration's mission of giving its students an edge in the competitive job market.

In high school, the administration aims to give a brief basis as to what robotics is and provide an introduction to the field in order to foster a greater interest in it. It desires a more focused curriculum in college to develop the in-depth skills that would allow graduates to apply their studies to industrial applications in the local community.

The administration is very excited to start such a program. The school has already made a presence in the robotics community by founding the annual Robotex National Robotics Technology Exposition, being hosted by OLLCF for the third time in July, 2009. With this conference, the school hopes to offer an environment that will excite the local community attendees and spark the interest of robotics in children. By putting forth a robotics curriculum, the school will also be offering a way to materialize this enthusiasm for technological advancement into concrete skill sets.

Faculty

The College of Computer Studies employs eight faculty members charged with instructing at both the high school and college levels. Four of these individuals, including its dean Gerry Lopez, hold master's degrees in related fields, two have bachelor's degrees, and the remaining two are TESDA-certified technicians. Each faculty member develops a complete semester-long syllabus and lecture schedule for the classes he or she teaches in accordance with the curriculum's written requirements. At present, the faculty shares the responsibility of teaching the high school computer classes due to the recent unexpected departure of a previous instructor.

The current college that is closest to engineering is the College of Maritime Studies. The curriculum is aimed at producing highly qualified midshipman. Therefore, the five faculty members have previously held positions at sea as technicians and operators as well as teachers and lecturers. Peter Lentze in particular has extensive industry experience as an operator, marine engineer, surveyor, and technician with many superintendent and management roles in his experience as well. He has extensive knowledge of the maintenance of industrial mechanical components, with particular respect toward ship and engine repairs.

Students

The student body is composed of 34 students in elementary school, 249 students in high school, and 999 students in college, of which 147 students are in the College of Computer Studies. The college students studying computer science cite the growing market demand and job prospects as an important factor in choosing their major. They have a strong motivation for entering technical careers, with ladderized ones like computer technicians enjoying great popularity across the country. Those high school students interested in science and technology similarly embrace their computer curriculum, and every high school student regardless of intended career path is taught the fundamental computer skills necessary for typical workplace environments.

Parents

As gathered from our interviews of 20 high school families, parents of OLLCF students seek a quality education that will offer better job prospects for their children. In fact, some specifically chose to enroll their children at OLLCF's high school because of the strong emphasis it places on technical fields such as computer science, a specialization other high schools in the area lack. Still, the cost of education is an important factor for many, and some parents are unsure of whether they can support their children through college, resulting in a majority of high school graduates not attending college altogether. Yet virtually all parents express a desire for their children to pursue high-paying careers in either emerging technical fields or traditionally popular Philippine jobs such as accounting or hotel and restaurant management.

SCOPE OF WORK

Our Task: Robotics Curriculum and Laboratory

One way to develop the technology infrastructure in Daet, Camarines Norte, Philippines through OLLCF is by incorporating cutting-edge topics, such as robotics, into the educational curriculum. This will further the college's goal of becoming an innovative educational institution in the Philippines and fulfill its mission of nurturing leaders who will advance the country technologically. OLLCF has asked us to include robotics topics in an introductory course for its high school and college as it currently does not have the capabilities to develop such programs itself.

The high school curriculum will offer the students a brief understanding of what robotics is and develop fundamental elementary robotics skills. At the college level, the robotics course will be offered first through the College of Computer Studies, and it could serve as a gateway to future programs introduced in the College of Engineering, which plans to open next year. The class will develop an in-depth understanding of robotics and provide critical skills necessary to empower the students to make a positive impact on technological industries.

In order to fit these courses into the existing curriculum, it is important that the faculty is familiar with the topics and capable of forming lessons around them. The College of Computer Studies dean, Gerry Lopez, currently teaches Modeling Simulation, a class that provides a nontechnical general overview of robotics, but otherwise the faculty has no experience directly dealing with the field. We therefore find it necessary to train them through a series of workshops.

In addition, it is critical for both levels of the robotics curriculum to include hands-on practical experience. This is best accomplished by having a robotics laboratory that functions as a dedicated working environment for these classes, whose hardware needs cannot fit in a typical classroom. In addition, the lab will serve as an entryway for future electronics and communications engineering and computer engineering workspaces. We will deliver a set of recommended equipment and a materials list for this lab.

Our Approach

Step 1: Evaluate Current Capabilities

One of OLLCF's desires is to eventually be at the forefront of robotics in the Philippines. As such, we must first begin with an investigation into how robotics will be helpful to them, and how we can fit it into their existing curriculum. Before creating any robotics classes, it is crucial to understand the current capabilities of the school in all relevant areas. This includes assessing the faculty, facilities and equipment, students, and the administration. By investigating the background of the faculty, we can identify key members that will participate with us in developing curricula that are sustainable once we leave. This will be accomplished as we review their curricula vitae, transcripts, and courses taught by them, as well as through individual conversations. It is also important to take a closer look at the existing Modeling Simulation class, providing us with a metric to gauge OLLCF's current understanding of robotics. This will include closely examining the existing hardware that the school possesses and auditing the course. Understanding the background of the students is essential as well, so we will look at the pace of classes, the material that can be covered in an academic period, the amount of knowledge retained, and the students' motivation. Administratively, we will investigate the general organization of courses taught in the computer science department both in high school and college.

Step 2: Analyze Alternatives

We will research all possible alternatives and baselines for an introductory course in robotics at the high school and college levels. This benchmarking analysis through online offerings and the only existing robotics class at OLLCF will provide us with alternatives that can be presented to the administration. One of our aims is to discover freely available lesson plans and teaching material that will provide subject matter that can currently be taught without extensive training of faculty members in unfamiliar topics. We will compare different curricula in each area we determine to be important. We will also identify positive components that can be taken from each individual curriculum and included in a compound curriculum that meets OLLCF’s needs. This process will involve presenting options for the administration to review and decide among according to its necessities. By the time we move to the next steps, we will have a direction for the curriculum and determine hardware that needs to be purchased for training purposes.

Step 3: Train Personnel

Once we agree with OLLCF on what set of robotics topics the curriculum should cover and what hardware will aid in that instruction, we will be able to develop a high level plan for the class. Having gathered the current capabilities of the faculty members, we will also be aware of what each instructor already knows and what foundational knowledge should be improved upon. From there, we will plan a series of daily workshops for the computer science faculty members intended to give the expertise necessary for teaching introductory robotics classes. Further detailed instruction that they seek for their own benefit will be included as much as possible. Following the conclusion of the workshops, we will offer recommendations for obtaining further training in order to teach more advanced classes. We will also propose a list of desirable skills and recommended experience to consider in hiring new faculty members.

Step 4: Develop Curriculum Cooperatively

Alongside training the personnel involved, our workshops will also offer the opportunity to collectively discuss methods for teaching robotics material, including demonstrations and lab exercises. This group environment further allows those parties involved to share any expected difficulties at each level of instruction and plan for questions that students taking these introductory classes might ask. Such activities will enable us to finalize the curriculum for an introductory class at both the high school and college levels in tandem with the faculty members and help those who will eventually teach the robotics courses develop their own lesson plans on the subject matter. This guarantees that each instructor is able to prepare a class which is consistent with his or her teaching style, thus ensuring the courses can be taught in a sustainable manner for future years.

Step 5: Propose Laboratory Setup

Initially, we will evaluate the current equipment in the maritime engineering and computer science departments to determine what OLLCF already has that can be included in a robotics laboratory. Afterward, we will recommend a complete setup for the lab with any other necessary equipment and materials that will enable students to gain hands-on experience in conjunction with the proposed curricula.

Expected Outcomes

We expect to develop one introductory robotics class for high school and another one for college, and the following table presents our expected outcomes at each step.

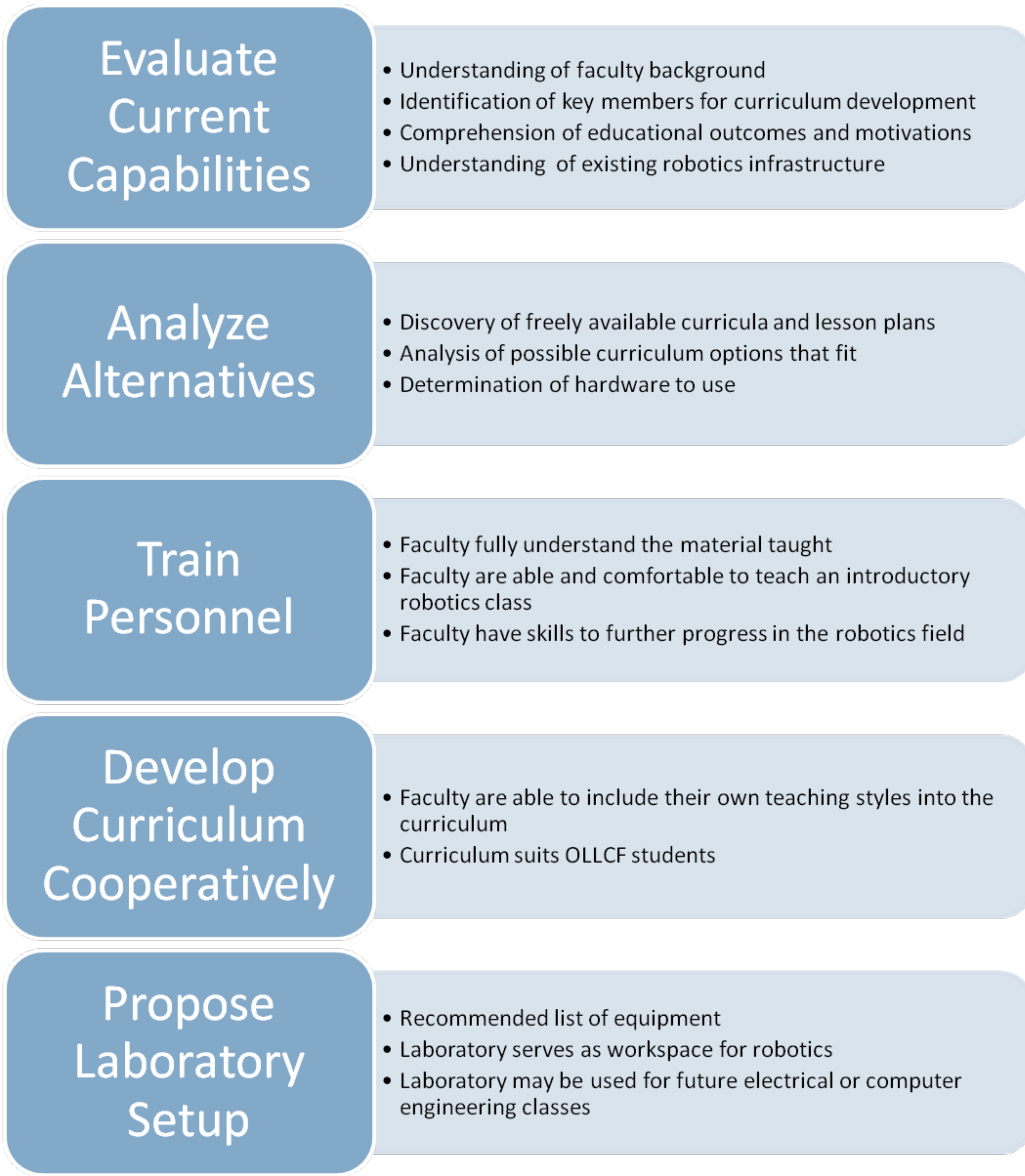


FIGURE 2: EXPECTED OUTCOMES

Additional Impact

The current computer science curriculum for high school students is likely to change depending on how involved OLLCF decides to make the secondary level introductory robots course. Without any modifications to other computer classes, there is only enough room for half of a year of robotics instruction during the fourth year of high school. If the administration prefers to give a more detailed course that lasts for an entire year, the material covered during the first three and a half years will have to be compressed or otherwise modified to fit within at most three years. From talking with computer science faculty who teach at the high school level, this is indeed feasible and even beneficial, as there is agreement that certain topics, especially those covered during the first year, take more time to teach than is necessary. Further compressing the curriculum into less than three years would allow the computer science faculty to add a greater emphasis on topics such as C and Java which themselves are beneficial to programming robots developed in the introductory robotics course.

Feasibility

Within the remaining time we have at OLLCF, training faculty members and collectively developing curricula with them can be accomplished for introductory robotics courses at both the high school and college level. As members of the computer science faculty—including its dean—have expressed great enthusiasm for participating in workshops and collaborative curriculum development, we anticipate that the introductory courses will succeed in giving students a solid foundational knowledge of robotics and will be sustained long after we leave.

While this encompasses everything that the administration desires for the high school, OLLCF seeks to eventually include upper level elective classes for college students pursuing a Bachelor of Science in Computer Science degree that would like a concentration in robotics. Developing curricula for these electives cannot be achieved within our ten-week time frame, and teaching such courses would largely be outside the capabilities of the instructors unless they were to pursue comparable advanced college courses or similar training themselves. Recommendations for such advanced training will be accomplished in Step 3, however, so it is perfectly possible for the faculty members to develop these upper level classes themselves in later years.

Work Plan

Evaluate Current Capabilities (Week 2)

- Understand content of relevant computer science and engineering courses
- Asses faculty background through curricula vitae and transcripts
- Asses students' programming skills obtained in computer science classes
- Audit computer science classes in high school and in college
- Investigate robotics applications in Philippine industry

Analyze Alternatives (Week 3)

- Survey other high school robotics curricula
- Survey other college robotics programs
- Analyze lesson plans and teaching material currently available
- Identify possible features that can be included in a curriculum in Step 4
- Determine hardware to use for training and courses

Train Personnel (Weeks 4–8)

- Hold daily workshops for one hour to teach relevant robotics topics to faculty involved
- Transfer resource and curriculum knowledge
- Recommend further advanced training
- Gather input and adapt curriculum as necessary

Develop Curriculum Cooperatively (Weeks 4–8)

- Meet after workshops for 30 minutes to co-develop curriculum
- Define desirable outcomes for robotics classes
- Develop complete curriculum for high school and college
- Create lesson plans and class outlines
- Recommend future goals and milestone curricula

Propose Laboratory Setup (Week 8)

- Evaluate current equipment in relevant departments
- Match proposed equipment to proposed curriculum

FIGURE 3: WORK PLAN

OUTCOMES AND RECOMMENDATIONS

Step 1: Evaluate Current Capabilities

Our investigation into Philippine industries shows that robotics, like many similar technical fields, does not yet has a large demand. Those areas where it would be most useful emphasize its practical applications such as in repair work and manufacturing. As such, we believe it would be best to include more hands-on learning rather than pure theoretical instruction when it comes to OLLCF robotics classes, especially at the introductory level where the most useful concepts are included for ladderized course purposes.

Given the lack of a College of Engineering, however, we believe it is beyond the current capabilities of the faculty to instruct students in every discipline related to robotics, particularly when it comes to constructing robots by oneself. Once the College of Engineering opens (plans exist to do so next year) with degree programs in mechanical engineering and electronics and communication engineering, later classes may be created which are tailored to the complete development of an entire robotics system, which we address in more detail in our Recommended Milestone Curricula section. For now, though, the only applicable college for a robotics program is the College of Computer Studies, and we thus believe that the introductory class included there should focus primarily on the programming aspects of robotics. This, we feel, fits best with the capabilities of the faculty and students. Additionally, doing so would enable OLLCF to build off of its only existing robotics class, Modeling Simulation, which is also programming-oriented as it is only available to students in the College of Computer Studies.

Having observed a few sessions of the Modeling Simulation class and even twice teaching its students some of the material from our workshops, we agree with OLLCF that the students have adequate preparation for an introductory robotics course focused on programming. Students were quickly able to adapt to the programming environment we demonstrated to them, and some teams even surprised us by coming up with novel solutions to our task after an independent investigation of the capabilities of the programming language, discovering functionality that we did not cover with them. Some of the success of the sessions we taught should also be attributed to the small group size we used, having students work in pairs. Addressed in detail in the Recommended Team Structure section, we feel such smaller groups allow for greater opportunities for individual learning. The way Modeling Simulation normally achieves this with the two robotics kits it currently has is by staging when each group may work on a given project, essentially taking turns with the kits rather than having everyone work on a task concurrently. We believe this should be avoided in the introductory robotics class we propose, as it drastically slows down the pace of the course, and it limits the number of hands-on robots students can build and program in the span of one semester.

In preparation for our workshops, we identified five faculty members to train and work with in developing the robotics curriculum. Three are instructors in the College of Computer Studies—Richard Dilan, Allan Timpac, and the dean, Gerry Lopez, who also teaches the Modeling Simulation course. Another, Salvador Calimbayan, is a graduate of electronics and communication engineering and the current OLLCF property custodian. Due to a lack of faculty members, he currently teaches mathematics and digital circuit classes, and is likely to continue as an instructor in the electronics and communication engineering department when it opens. The last is the acting principal of the high school, Maria Theresa Malaborbor. Although she has no background in related technical fields, we thought including a high school instructor would be necessary in determining appropriate activities for the secondary level robotics class.

Step 2: Analyze Alternatives

This section presents all of the analysis and benchmarking of alternatives that we have been considering to build up a robotics curriculum for OLLCF. Tables 1 and 2 show all of the educational resources being considered, given in a format demonstrating what other universities and high school have already developed. It is organized to evaluate each alternative in the following categories: estimated level of difficulty for OLLCF's students (scales 1–10, with 10 being the hardest), length of the class, format of the material, and hardware kit used. A second section evaluates the topics covered in the mechanical engineering, electrical engineering, and computer science disciplines. It also includes additional comments and features that we think are beneficial to include in OLLCF's future curriculum. All of these (mostly free) materials will be used as the building blocks for OLLCF's curriculum as we work together to merge all these alternatives in creating classes that will be sustainable in the long run.

The hands-on curricula we found that included more than just theory used one of five commercially-available robotics kits during the classes: LEGO Mindstorms RCX, LEGO Mindstorms NXT, VEX Robotics, iRobot Create, and E-Gizmo's Robo-box. To determine the appropriate fit for OLLCF, Table 3 evaluates each of these on six criteria: price, development support, programming language support, -availability of teaching material, first-party hardware, and capability of interfacing with third-party hardware.

Curriculum Alternatives

TABLE 1: CURRICULUM COMPARISON PART 1

	PMHS Robotics	CMU Robotics Academy (LEGO)	University of Colorado's Class	Proactive PD	Robo Works	CMU Robotics Academy (VEX)
Link or Source	http://www.pittsfield.k12.nh.us/HS_Subjects/Science/Robotics/default.html	http://www.education.rec.ri.cmu.edu/roboticscurriculum/index_to_robotics.htm	http://csel.cs.colorado.edu/~bauerk/legorobots/Default.htm	http://www.proactivepd.com.au/index.php?p=1_11	http://www.robo-works.net/roboeducators.html	http://www.education.rec.ri.cmu.edu/roboticscurriculum/vex_online/
Difficulty (1–10)	4	5	6	4	8	7
Length of Course	36 weeks	Estimate of 40 hours of class	15 weeks	6 weeks	30 weeks (five 6-week units)	Estimate of 40 hours of class
Material Format	Class discussion, students keep an engineering journal per lab, and labs are held in groups.	PowerPoint presentations, worksheets, labs.	HTML overview of each lecture section. Most work consists of labs where teams build a robot for a final competition.	Students will work in small groups (2 or 3). Designed for students to be hands-on to solve several sub-tasks. At the end of each task, there is a presentation part to teach communication skills.	Week-by-week syllabus breakdown with material recommended for each day of the week. Material includes lecture and handout subjects, activities, demos, homework.	Use of VEX kit, written assignments, quizzes and exams on basic theoretical concepts, oral presentations, and expositions.
Kit	LEGO RCX with ROBOLAB ($200, plus $100 for ROBOLAB).	LEGO Mindstorms RCX hardware and ROBOLAB software ($300).	LEGO Mindstorms RCX and custom robotics board with additional miscellaneous hardware provided by teams ($750 plus up to $20 spent by teams).	LEGO Mindstorms NXT education set ($250).	Theory with miscellaneous hardware needed for lab demos (no kit used).	VEX kit ($300–$750), plus additional cost for the curriculum and unlimited licenses for the VEX software ($500).

	PMHS Robotics	CMU Robotics Academy (LEGO)	University of Colorado's Class	Proactive PD	Robo Works	CMU Robotics Academy (VEX)
MechE	Gears (ratio and trains), rotary-linear motion, assemblies, tension, trusses, compression.	Gears, speed, torque, simple machines.	Forces, friction, torque, simple machines, inertia, springs, rubber bands, counter weights, gears.	Teach spatial awareness of robots to understand their operation, movable structures, motors, servos, counter weight.	Drive trains, gears, sprockets and motorsheat engines, power transfer system, pneumatics.	Motors, torque, train, gears, gear ratio, RPM, calculate power of a system, forces, energy. Dedicated module for these topics.
EE	Sensors.	Hardware, electronic control, sensors (touch, light, temperature, rotational).	Charge, voltage, current, batteries, circuit elements, motors, pulse width modulation, sensors (as transducers, tactile, light, other).	Hands-on understanding of line follower sensors, range sensors, motors. Practical labs include Four Poles Challenge, line following, Sumo Challenge, robot gripper.	Basic circuitry, motor control, generators, sensors, solid state, robot control.	Explanation of how all sensors work such as encoders, line detectors, switches, color sensors, etc. There is a dedicated module for this section.
CS	Programming with ROBOLAB, obstacle avoidance algorithms, brick communication.	Intro to advanced programming (variables, timers, multitasking, syntax-based language).	Uses Interactive-C, assumes no prior programming experience.	Use of programming environment provided by kit. Students understand practical applications of math like numbers and fractions in programming.	Basic introduction (looping, I/O), assumes no prior programming experience.	Program behaviors and simple algorithms in the VEX environment. There is a dedicated module for programming.
Business	Engineering journal, MLCAD to communicate designs.	Project management.	Timeline planning.	Design process, documentation of major steps, changes, and upgrades.	CAD.	Project management, safety, design skills.
Additional Comments	Very inclined in applied physics. Includes a project on building a bridge.	Touches major topics in robotics in 8 units with practical labs. Developed by CMU.	Focuses on team design and implementation to perform a few labs leading up to a final game competition amongst teams.	Tasks are open-ended to encourage creative designs and teach applied solving skills. There is encouragement for students to participate in FIRST competition.	Very detailed lesson plans of comprehensive material.	Service to review curricula with CMU faculty. Expense can be a problem. Unlike LEGO (which looks like a toy), VEX robots are very close to what "real" robots are. Stability is great.
Usable Parts	Could potentially use their project ideas and layout of topics, as well as borrowing engineering journal.	Gears, speed, torque, simple machines, basic hardware, programming, and electronic concepts.	Basic mechanics, motors, basic electronics, sensors, controls, programming.	Project ideas on line follower, sumo competition, robot grabber. Communication skills developed through student presentations at the end of each major lab.	Mechanics, pneumatics, electronics, programming.	Not enough material to ake usable parts without urchasing curriculum, but an look for sources on how ach sensor works and how o calculate power necessary or driving any given robot.

TABLE 2: CURRICULUM COMPARISON PART 2

	OLLCF	Java Robotics	Stanford's Robotics Class	Kiss Institute for Personal Robotics	General Robotics	NASA Robotics Education Project
Link or Source	OLLCF's curriculum (see Appendix A).	http://www.ridgesoft.com/articles/education/education.htm	http://www.stanford.edu/class/cs223a/	http://www.kipr.org/curriculum/content.html	http://generalrobotics.org/	http://robotics.arc.nasa.gov/edu/lessons/autonomy.htm
Difficulty (1–10)	3	7	10	9	10	9
Length of Course	18 weeks	15 lessons	9 weeks with 2.5 hours per week	8 activities	15 weeks	10 exercises
Material Format	Lecture, quizzes, labs.	Course outlines, lesson plans, online tutorial, programming assignments, building activities.	Class handouts available online, as well as videos of the lectures: http://www.youtube.com/view_play_list?p=65CC0384A1798ADF	Proposed assignments and labs.	Homework assignments, practical labs, exams, complete course outline.	Exercise descriptions.
Kit	Robo-box ($200 per kit), use of Logo programming language.	Custom kit.	Theory (no kit used).	LEGO Mindstorms as basis.	Theory and labs (no kit used).	Various kits can be used, just provides an outline of possible projects for students to construct robots to do.

	OLLCF	Java Robotics	Stanford's Robotics Class	Kiss Institute for Personal Robotics	General Robotics	NASA Robotics Education Project
MechE	Basic motors, wheels, and structural components.	Chassis, motor, wheel.	Geometry, kinematics, statics, dynamics, control.	Gears, motors, forces, torque.	Kinematics, dynamics, control.	Building actual robots.
EE	Basic sensors (touch, light, infrared).	LCD, sensors, power source, serial cable.	None.	Sensor activities with photodiodes, infrared, range, bump/touch, shaft encoders including computer programming code, power.	Sensors, interface with MIT Handyboard.	Building actual robots.
CS	Uses Logo programming, a simple graphical language of drag and drop icons for actions.	Java programming, testing, debugging, maneuvering, behavioral-based controlling, line following, sensing, tracking.	None.	Programming introduction, Interactive-C tutorials.	C programming, algorithms for path and motion planning, computer vision.	Building actual robots, path planning, practical parts.
Business	None.	None.	None.	None.	None.	None.
Additional Comments	This course is a very basic overview of robotics. It is very limited due to the kit used, which is targeted for younger audiences, not college students.	Emphasis on Java programming and interfacing.	Theory only of robotics mechanics. Way outside scope of OLLCF.	Very detailed instructions regarding teaching (reads like a learning site or tutorial as oppose to proposal of curriculum site). Also explains on the website how different things work and the theory of why they work. Good basis for teaching if the instructors are not experts.	May be too much for OLLCF, complicated topics (especially when going into 3D vectors for all the motion planning algorithms).	Just possible project descriptions.
Usable Parts	Unit on introduction to robotics.	Programming.	Path planning, spatial representations and models.	Gears, torque, force, motors, sensors, and programming.	Outside of scope, maybe useful in the future.	Project ideas.

Recommended Curriculum Approach

As seen from our comparison chart, there exist many free available resources. The ones displayed are some of the most relevant to OLLCF's objectives in robotics. Although no single curriculum shown will exactly fit the school, we recommend following that of the Carnegie Mellon University Robotics Academy using LEGO Mindstorms NXT in addition with lesson plans, topics, and exercises from all other curricula, especially those that employ our recommended hardware kit. As our workshops progress, however, we will be working with OLLCF faculty to develop a custom curriculum tailored to the college and high school. The reasons for selecting the CMU Robotics Academy curriculum as a guide are explained for each category:

- The level of difficulty is appropriate at a range go 4-6 to match the students' background.
- The estimated length of the course is shorter than a semester, allowing for further topics to be introduced or existing ones to be reinforced with additional material and exercises.
- The format of the material allows for easy adjustments according to the progress of the faculty workshops.
- It provides a good overview of all topics that should be covered in an introductory robotics class for students lacking a strong background in programming and the sciences.

Hardware Kit Alternatives

TABLE 3: HARDWARE KIT COMPARISON

Hardware	Price (USD)	Development Support	Programming Language Support	Availability of Teaching Material	First-Party Hardware	Capability of Interfacing with Third-Party Hardware
LEGO Mindstorms RCX	$200	LEGO-supported online development community, which is one of the largest for hobbyist and educational robotics. Third-party organizations such as FIRST also offer development support.	LEGO-supported RCX Code (graphical drag and drop) and ROBOLAB (based on LabVIEW). Also third party-support of variations of C, C++, Java, Forth, Lua, Visual Basic, and Lisp.	Plenty of online resources, printed material, and books. It also has a curriculum developed by the Robotics Academy at CMU. Several colleges and high schools across the US use LEGO as a preferred kit.	Supports construction with all LEGO pieces, coming with 200+ such parts. Kit also comes with 2 DC motors, 1 light sensor, and 2 touch sensors. Renesas-based 8-bit microprocessor supports 3 inputs and 3 outputs at any one time.	None supported. However, custom sensor tutorials exist, and they can made by those with knowledge of electronics.
LEGO Mindstorms NXT	$250	LEGO-supported online development community, which is one of the largest for hobbyist and educational robotics. Third-party organizations such as FIRST also offer development support.	LEGO-supported NXT-G, which is a graphical drag and drop interface and is based on LabVIEW. It also supports third-party variations of Ada, assembly, C, C++, Java, MATLAB with Simulink, Lua, and Ruby.	Plenty of online resources, printed material, and books. It also has a curriculum developed by the Robotics Academy at CMU. Several colleges and high schools across the US use LEGO as a preferred kit.	Supports construction with all LEGO pieces, coming with 500+ such parts. Kit also supports Bluetooth technologies and comes with 3 servo motors, 1 ultrasonic proximity sensor, 1 light sensor, 1 touch sensor, and 1 sound sensor, with additional sensors (temperature, accelerometer, gyroscope, RFID reader) commercially available. ARM-based 32-bit microprocessor supports 4 inputs and 3 outputs at any one time.	Firmware is open source, and LEGO offers developer kits for software, hardware, and Bluetooth, enabling the creation of and full interfacing with custom programming environments, sensors, or applications and Bluetooth devices.

Hardware	Price (USD)	Development Support	Programming Language Support	Availability of Teaching Material	First-Party Hardware	Capability of Interfacing with Third-Party Hardware
VEX Robotics	$300–$750	VEX-supported online development community with 2,000 active members. Third-party organizations such as FIRST also offer development support.	If the kit purchased does not contain the VEX microcontroller, robots can only be controlled through a radio remote control. Otherwise can use programmed in variations of C.	There is a much smaller community of hobbyists and developers, and there are fewer resources compared to LEGO Midstorms since VEX was created in 2006. As with LEGO, the Robotics Academy at CMU also has created a curriculum using this system.	Supports construction with all metallic parts. The basic starter set comes with the VEX controller that supports 16 I/O ports for analog and digital signals. It includes 2 bumper switches, 2 limit switches, and 4 motors. There are various VEX expansion kits—drive, pneumatics, advanced sensors (includes encoders, light sensors, limit switches, buttons, infrared sensors, range sensors)—available for $250+ each.	As of now, interfacing with VEX microporcessor is very limited. However, there is current development to work with the Qwerk board. The expected release will be in a year called VEX Pro. This would allow what the LEGO NXT can do (and even more), since the Qwerk board is based on a AVR microprocessor that is open source.
iRobot Create	$200–$300	iRobot-supported online development community which is smaller than those of VEX and LEGO.	iRobot's Command Module supports C and C++. Third-party microcontrollers may support additional languages.	Material not widely available, although a few educational institutions do use the system for classes and research, including CMU, Simon Fraser University, and Georgia Institute of Technology.	None available other than the base robot and iRobot's Command Module.	iRobot Command Module is very limited, but any microcontroller with a serial interface can be used. Additional hardware interfaces are therefore dependent on the third-party controller being used., with the Gumstix line being a popular choice.
E-Gizmo's Robo-box	$175	None available.	The microcontroller only supports a graphical programming language called Logo (similar to the one used in LEGO Mindstorms) and Cricket, a simple text-based language.	None.	Uses the i-Box III microcontroller with 16 KB memory, contains a chassis with mounting plates and assembly hardware. Also has a very simple wheel track drive and gear systems. It has 2 DC motors, 2 touch sensors, 2 light sensors, and 2 infrared proximity sensors.	No existing third-party interfacing, but could probably be hacked by someone knowledgeable in electronics.

Recommended Hardware Kit

After analyzing all of the hardware alternatives to use, we recommend the LEGO Mindstorms NXT, as it stands out as a clear winner in every category:

- Its price is competitive, only higher than the older and more obsolete LEGO Mindstorms RCX (by $50) and the very limited E-Gizmo Robo-box set (by $75). Furthermore, with the release of version 2.0 of the NXT set this upcoming August, the cost of the current 1.0 version could drop, lessening these price gaps further.
- With the LEGO Mindstorms brand existing since 1998, the NXT also boasts the largest online development community.
- It supports variations of every programming language that the non-LEGO sets offer, as well as some common languages that are exclusive to the NXT kit (Java, MATLAB, and Ruby).
- As the most widely used set, the NXT has the most existing curricula built around it. Its popularity in high school and college courses alike underscores its prominence when it comes to educational robotics.
- It is the only set to provide a sound sensor, while the other sensors that come with it offer the same functionality as those in every other kit. Additionally, the variety of third-party sensors approved by LEGO adds even greater capabilities for those who want them.
- The NXT microprocessor is the only currently-available open source option, and LEGO developer kits make interfacing with various software, hardware, and Bluetooth devices extremely simple. This invaluable feature allows the LEGO sets to be used for future advanced classes as well by creating and adding additional interfaces such as vision.

Additionally, the LEGO Mindstorms NXT requires very minimal skill and prerequisite knowledge to use (far less than that necessary with the VEX and iRobot Create systems), and yet it is highly versatile and can be used for even complex robots (a capability the Robo-box set lacks). While similar in ability to the RCX system, the NXT is obviously newer and has the same functionality as its predecessor, plus more. For all these reasons, we decided that the correct choice for OLLCF's robotics courses is the LEGO Mindstorms NXT.

Recommended LEGO Mindstorms NXT Version

Numerous versions of the LEGO Mindstorms NXT set are commercially available. Each one has slight differences in the number of construction pieces and types of sensors. While these differences are minor enough that they will not affect the capabilities of robots built using the set, they do have an impact on the price and what additional materials OLLCF would need to buy.

The version discussed in the previous comparison with other hardware kits is the most commonly used set, the retail NXT 1.0. Starting August 1, 2009, the LEGO Group has also released the NXT 2.0 version of this set. The most drastic chances that come with the upgraded version are a replacement of the light sensor with a color sensor and the replacement of the sound sensor with a second touch sensor. In contrast with the retail sets, the Education NXT Base Set contains more sensors and material that is useful in a classroom setting, although at the expense of fewer construction pieces. To offset this, the LEGO Group also sells a companion Education Resource Set that consists of additional pieces to supplement the Education NXT Base Set.

The table below presents a comparison of the retail NXT 1.0 set, retail NXT 2.0 set, Education NXT Base Set, and Education NXT Base Set plus Education Resource Set in the important areas that should be considered when choosing one version.[2]

TABLE 4: LEGO MINDSTORMS NXT VERSION COMPARISON

	Retail NXT 1.0	Retail NXT 2.0	Education NXT Base Set	Education NXT Base Set + Education Resource Set
Price (USD)	$250	$280	$280	$360
NXT Intelligent Brick	1	1	1	1
NXT Servo Motors	3	3	3	3
NXT Touch Sensor	1	2	2	2
NXT Light Sensor	1	1 (Color Sensor)	1	1
NXT Sound Sensor	1	0	1	1
NXT Ultrasonic Sensor	1	1	1	1
NXT Converter Cables (to connect to older RCX sensors)	0	0	3	3
Lamps	0	Color Sensor can function as lamp	3	3
NXT Rechargeable Battery and NXT Charger	0	0	1	1
Total Pieces	577	619	431	1102
Storage Boxes	0	0	1	2

A summary of the pros and cons of each of these four options is presented below.

RETAIL NXT 1.0 SET

Pros:

- Cheapest
- Reasonable number of pieces to build larger robots

Cons:

- No rechargeable battery pack

---

[2] For further discussion on the differences between the retail NXT 1.0 set and the Education NXT Base Set, see http://www.nxtprograms.com/help/parts/9797.html.

RETAIL NXT 2.0 SET

Pros:

- Extra touch sensor
- Light sensor is upgraded to color sensor
- More pieces to build bigger robots

Cons:

- No sound sensor
- No rechargeable battery pack

EDUCATION NXT BASE SET

Pros:

- Extra touch sensor
- Rechargeable battery pack
- Storage bins to organize pieces

Cons:

- Fewer pieces to construct robots, so some larger robots may not be possible

EDUCATION NXT BASE SET PLUS EDUCATION RESOURCE SET

Pros:

- Extra touch sensor
- Rechargeable battery pack
- Storage bins to organize pieces
- Most number of construction pieces

Cons:

- Most expensive

We recommend buying the retail NXT 1.0 version and using rechargeable AA batteries to solve the issue of constant battery usage. This remains the cheapest option while still offering a great deal of flexibility in which robots can be constructed. Furthermore, purchasing storage bins similar to those that come with the Education NXT Base Set would assist in maintaining each separate set and minimizing the loss of pieces.

Recommended Number of LEGO Mindstorms NXT Sets

Once a LEGO Mindstorms NXT version is chosen, OLLCF's administration must also determine the number of such sets to purchase. The bare minimum number of sets is 5, which would translate to students working in groups of 6–7 in a class size of 30.

A better number to buy, however, is 7. This would result in groups of 4–5 students, or alternatively groups of 5 students while one set is dedicated to faculty use for planning lessons.

The ideal number of sets to purchase is 9. This ensures no more than 4 students per group (LEGO Education actually recommends only 2 or 3 students working in a group) with one set dedicated to the faculty.

Step 3: Train Personnel

Our workshop sessions were held every day for four weeks (for details on the workshop syllabus, see Appendix B). Formal workshop settings lasted one week shorter than originally planned due to the faculty members involved becoming increasingly busy with the Robotex National Robotics Technology Exposition, which OLLCF hosted during the planned fifth week. As such, we continued individual training afterward with those interested in order to make up for the lost week of instruction. By the end of the workshops, those faculty members who were able to consistently attend developed a strong foundation in the basic topics underlying robotics that would invariably be included in an introductory course.

Recommended Qualifications for New Hired Faculty

In teaching basic robotics topics during the workshops, we discovered a number of qualities that proved invaluable in instructing large groups of students on the material. OLLCF should look for these qualifications when hiring new faculty members involved in its robotics program, whether as teachers or researchers. These are broken down by applicable department.

The following are recommendations of qualifications for new employees hired in the computer science department:

- Well-versed in computer programming logic structure
- Knowledgeable in the C and Java languages
- Skilled in debugging and troubleshooting

The following are recommendations of qualifications for new employees hired in the electronics and communication engineering department:

- Knowledgeable of different sensors and how they work
- Knowledgeable of analog and digital circuits
- Understands electrical controls and components
- Has foundation in feedback control systems

The following are recommendations of qualifications for new employees hired in the mechanical engineering department:

- Has a thorough foundation in force analysis, stress analysis, material analysis, and dynamics
- Knowledgeable of mechanical controls and mechanical components

The following are recommendations of qualifications for new employees which are applicable to all departments:

- Critical thinker
- Detail-oriented
- Has good communication skills and can work effectively on a team to create a robot
- Has a thorough math and physics background
- Ability to approach a problem systematically
- Ability to breakdown a complex task into smaller more manageable parts (particularly useful in troubleshooting and debugging a robot)

Step 4: Develop Curriculum Cooperatively

Overview

The robotics classes and curricula resulted from a co-development with OLLCF's administration and faculty. As we worked with the administration, we gathered important information that guided us in understanding the school's goal of becoming a Center of Excellence in robotics. With this, we provide a recommendation on milestone curricula that offer general directions for the school to follow according to different levels of advancement specified by eight phases.

Within the immediate future, we have set up the first phase for the school as we worked closely with OLLCF's faculty to develop the first relevant robotics classes. For this purpose, we held workshops to train the faculty on introductory topics in robotics through lectures and practical activities using LEGO Mindstorms NXT sets. The teaching materials and topics covered were obtained from our alternatives analysis and selected in accordance with the results from our evaluation of the school's current capabilities. Given that the school already has capacity in the computer science area, the workshop activities emphasized programming skills. The structure of the workshops encouraged a discussion session at the end that allowed teachers to decide:

- Whether the material would be suitable to students
- Whether more material should be included in a topic
- What would be effective methods of teaching the class
- What organizational system would be the best to organize lab sessions and topics
- How he or she would teach the classes
- How long the topic would take to cover in college and in high school

As the workshops progressed, the curriculum began to take form as we defined how the lab activities would be best managed. As a result, we created roles and lab guidelines that initiated from these discussions with the teachers. The faculty reacted positively to the idea, so we expanded from it and created solutions following the lab guidelines for each of the practical activities held in the workshops (see Appendix D). Nevertheless, during the initial stages of the workshop, we encountered difficulty in teaching the concepts of programming logic for a robot. Consequently, we decided to apply the Human Robot Game (see Appendix F), which is an activity taken directly from one of our resources. In addition, we decided to organize the classes to introduce each block of the NXT-G programming environment in a specific order (see NXT-G Programming Sequence for Labs). According to the attendants of the workshop, this would help students maximize the learning of the programming environment by building up the knowledge step by step.

College Syllabus

See NXT-G Programming Sequence for Labs to consult with programming objectives achieved in each lab activity and Appendix D for solutions and more details on the labs.

I. COURSE NO

CS14

II. COURSE TITLE

Introduction to Robotics

III. COURSE DESCRIPTION

This course attempts to provide students with the basic mechanical and programming knowledge necessary to build a robot using LEGO Mindstorms NXT sets. Topics covered include basic control theory, mechanical concepts, sensors, flow charts, design process, and programming. In addition, students will build basic robots to introduce these topics.

IV. GENERAL OBJECTIVES

1. To know the basic components needed to build a robot
2. To gain an understanding of basic robotic controls
3. To apply theory from lectures to build robots following a design process
4. To assume team roles from a engineering design perspective
5. To understand robotics programming logic
6. To use methods for troubleshooting and debugging robots
7. To develop the ability to critically analyze problems by breaking them down into smaller and more manageable steps
8. To develop the ability to analyze logic and actively seek out different solutions

V. TERMS

1 Semester

VI. COURSE CONTENT

No. of hours	Contents	Instruments, Materials, and Methods	Evaluation
1	▪ Introduction to robotics ▪ Closed-loop feedback control system ▪ Orientation for lab and roles	▪ PowerPoint Presentations ▪ Whiteboard ▪ Lecture ▪ Class Discussion	▪ Assignments ▪ Quiz ▪ Recitation
7	▪ Introduction to sensors ▪ Basic mechanics ▪ LEGO components and sensors ▪ Human Robot Game ▪ Introduction to LEGO programming ▪ Lab: building Tribot following LEGO software: ○ Tribot base ○ Tribot bumper ○ Tribot grabber ○ Tribot sound sensor ○ Tribot light sensor ○ Tribot ultrasonic sensor	▪ PowerPoint Presentations ▪ Whiteboard ▪ Lecture ▪ Class Discussion ▪ LEGO Mindstorms NXT sets ▪ Computers	▪ Presentations ▪ Quiz ▪ Exam ▪ Recitation ▪ Assignments
12	▪ Bumper Robot ○ Drive straight until bumped ○ Turn 180 degrees ○ Drive back at twice the power ○ Use rotation sensor to stop at initial point ▪ Object Follower ○ Find an object ○ Follow the object ○ Display "_ cm from object" ○ Keep a constant distant from object ▪ Line Follower ○ Follow a black line ▪ Photovore or Phototrope (light follower or avoider)	▪ Whiteboard ▪ Lecture ▪ LEGO Mindstorms NXT sets ▪ Computers	▪ Presentations ▪ Design notebook content check (Robot Layout Diagram, Logic Flow Chart, steps followed) ▪ Lab exam

	○ Follow or avoid light ○ Play a tone according to light intensity		
16	▪ Stopwatch ○ Wait for NXT button press ○ Count time since button press ○ Display " __ seconds" ○ Use stop block ▪ Clapper ○ Calibrate ○ Count number of claps and beep that many times ○ Beep at the intensity of the most recent clap ○ Move number of times that was clapped ○ Show use of sensor loop controlled by timer ○ Use sensor blocks and logic switch ○ Stress that sensors are good for output values ▪ Random Numbers ○ Play sound according to range of number ▪ Combine ○ Press touch sensor to combine previous activities ○ One = run Stopwatch ○ Two = run Random Numbers ○ Three = run Clapper ○ Display options accordingly	▪ Whiteboard ▪ Lecture ▪ LEGO Mindstorms NXT sets ▪ Computers	▪ Presentations ▪ Design notebook content check (Robot Layout Diagram, Logic Flow Chart, steps followed) ▪ Lab exam
15	▪ Text Messenger ○ Bluetooth technology ○ Send Message and Receive Message blocks ▪ Differential Drive ○ Overview of PID control ○ Use of motor blocks ○ Relationship between linear distance traveled, wheel rotations, and wheel circumference ○ Calculation of speeds to achieve desired motion ○ Smooth turning ▪ Obstacle Avoidance ○ Reactive versus predictive behaviors in robots	▪ Whiteboard ▪ Lecture ▪ LEGO Mindstorms NXT sets ▪ Computers	▪ Presentations ▪ Design notebook content check (Robot Layout Diagram, Logic Flow Chart, steps followed) ▪ Lab exam

	○ Difference in detection and reaction to near and far obstacles		
3	▪ Design and Create Own LEGO Robot ○ Good engineering practices ○ Good programming practices	▪ Whiteboard ▪ Lecture ▪ LEGO Mindstorms NXT sets ▪ Computers	▪ Presentations ▪ Design notebook content check (Robot Layout Diagram, Logic Flow Chart, steps followed)

REFERENCES

Power Point Presentations:

- Introduction to Robotics
- Orientation for Roles and Lab Activities
- Introduction to General Sensors
- LEGO Sensors and Actuators
- Basic Mechanics

Websites:

- http://www.education.rec.ri.cmu.edu/roboticscurriculum/index_to_robotics.htm
- http://www.proactivepd.com.au/index.php?p=1_11
- http://www.owlnet.rice.edu/~elec201/Course_Notes.htm
- http://www.smartred.eq.edu.au/Science%20of%20Robotics/Main%20Page.html
- http://www.legoengineering.com/resources-a-downloads-mainmenu-60.html

Software:

- LEGO Midstorms NXT v1.1

High School Syllabus

See NXT-G Programming Sequence for Labs to consult with programming objectives achieved in each lab activity and Appendix D for solutions and more details on the labs.

Subject : Computer 4

Subject Description : Introduction to Robotics

Time Allotted : 1 hour and 20 minutes (MWF) - 4 hours per week

Credit Unit/s : 2 units

VISION

The High School Department of Our Lady of Lourdes College Foundation envisions to serve as a strong bridge between elementary and college life. As a transition period of the educational maturity of the youth, our department shares an active role in the pursuit of knowledge, wisdom, and truth for total development and formation of the youth.

MISSION

It is the mission of the High School Department to provide quality education and upright training to every student to prepare him to become a responsible and dependable member of the family, the community, and the world.

OBJECTIVES

1. To produce secondary school graduates who possess the qualities of life-long learner
2. To provide a learning environment where all students are treated equally
3. To prepare students to be vigilant and alert in assuming roles and responsibilities as young citizens in the school and community
4. To inculcate in every student the values of great Filipinos worthy of emulation
5. To instill in every student the development of skills and competence in mathematics, science, and communication to become competitive and productive citizens in the current millennium

BRIEF DESCRIPTION

This course attempts to provide students with the basic mechanical and programming knowledge necessary to build a robot using LEGO Mindstorms NXT sets. Topics covered include basic control theory, mechanical concepts, sensors, flow charts, design process, and programming. In addition, students will build basic robots to introduce these topics.

GENERAL OBJECTIVES

1. To know the basic components needed to build a robot
2. To gain an understanding of basic robotic controls
3. To apply theory from lectures to build robots following a design process
4. To assume team roles from a engineering design perspective
5. To understand robotics programming logic
6. To use methods for troubleshooting and debugging robots
7. To develop the ability to critically analyze problems by breaking them down into smaller and more manageable steps
8. To develop the ability to analyze logic and actively seek out different solutions

Time Frame	Topics	Objectives	Competencies	Contents	Teaching Strategies	Materials	Evaluation
First Quarter	Introduction to Robotics	▪ Understand general concept of robotics and its multi-disciplinary nature ▪ Understand closed-loop feedback in a basic Robot Layout Diagram ▪ Realize the importance of different roles in a robotics team	▪ Recognition of each step followed in the lab guidelines	▪ What is a robot? ▪ What if I was a robot? ▪ Closed-loop feedback control system ▪ Different fields that encompass robotics ▪ Roles on a team: engineer	▪ PowerPoint presentation ▪ Lecture ▪ Class discussion	▪ Projector ▪ Handout for lab guideline and roles ▪ Whiteboard	▪ Quizzes ▪ Recitation ▪ Homework

			assignments, checkpoints			
Introduction to Sensors and Actuators	▪ Understand how robots perceive and interact with the real world ▪ Understand what components make a robot move ▪ Differentiate what is an input and an output for a robot	▪ Intuition of how robots perform sensing and moving	▪ What makes a machine a robot? ▪ What is sensing? ▪ Why do robots need sensors? ▪ Common sensors ▪ What is an actuator? ▪ Common actuators ▪ How do sensors and actuators fit in the close-loop feedback control system? ▪ Human Robot Game	▪ PowerPoint presentation ▪ Lecture ▪ Class discussion	▪ Projector ▪ Whiteboard	▪ Quizzes ▪ Recitation
LEGO Sensors, Actuators, and Components	▪ Understand difference between analog and digital data ▪ Understand difference between analog and digital sensors ▪ Identify each of the LEGO sensor and actuator features	▪ Ability to compare and contrast between analog and digital data and sensors	▪ Analog versus digital data ▪ Demonstration of the touch, sound, light, and ultrasonic sensors from the "Try Me" menu on the NXT brick ▪ Different measurements taken from each sensor ▪ Analog and digital LEGO sensors	▪ PowerPoint presentation ▪ Lecture ▪ Class discussion ▪ Demonstration of LEGO sensors and motors	▪ Projector ▪ NXT brick with all sensors and motors ▪ Whiteboard	▪ Quizzes ▪ Recitation ▪ Hands-on evaluation
Basic Mechanics	▪ Comprehend Newton's laws ▪ Define torque ▪ Familiarize self with simple machines (levers, pulleys, and gears)	▪ Skills on applied physics and mechanics	▪ Newton's laws of motion ▪ Torque ▪ Mechanical advantage and its applications ▪ Levers ▪ Pulleys	▪ PowerPoint presentation ▪ Lecture ▪ Demonstration	▪ Projector ▪ LEGO sets ▪ Whiteboard	▪ Exercises ▪ Solving problems ▪ Recitation

				▪ Uses and different types of gears	reporting		
	Introduction to LEGO Programming Environment	▪ Be able to navigate through NXT-G editor ▪ Manage profiles ▪ Save and open files	▪ Memory	▪ Copy, paste, undo ▪ Moving blocks around and readjusting sequence beam ▪ Navigation of editor ▪ Opening and creating new profiles ▪ Saving and opening files ▪ NXT brick status ▪ Downloading programs	▪ Lecture ▪ Demonstration of LEGO programming environment	▪ Projector ▪ Computer with LEGO software installed	▪ Quizzes
	Tribot	▪ Know how the closed-loop feedback system fits in each of the robots ▪ Apply previously learned knowledge ▪ Learn all of the functions of the wait block ▪ Follow lab guidelines and roles	▪ Development of flowchart to understand logic processes ▪ Ability to follow assembly instructions	▪ Tribot base ▪ Tribot bumper ▪ Tribot grabber ▪ Tribot sound sensor ▪ Tribot light sensor ▪ Tribot ultrasonic sensor	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Written exam about lab guidelines and roles ▪ Oral quizzes about the function of each block learned
Second Quarter	Bumper Robot	▪ Use touch sensor in a lab setting ▪ Know how rotation sensors work and their applications ▪ Follow lab guidelines and roles	▪ Identification of a dynamic system on a robot	▪ Reinforce idea of inputs and outputs in a closed-loop feedback control system ▪ Different ways of steering a robot ▪ Touch sensor ▪ Rotation sensor ▪ Move block	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum

				▪ Loop block (loops until sensor data received)			
	Object Follower	▪ Use ultrasonic sensor in a lab setting ▪ Know how to display text and numbers ▪ Follow lab guidelines and roles	▪ Basic control flow logic for programming ▪ Intuition of how robots perform sensing and moving	▪ Reinforce how robots sense and then act step by step ▪ Basic control flow using infinite loops ▪ Switch blocks using sensor control ▪ Ultrasonic sensor ▪ Display images, text, numbers ▪ How inputs and outputs work in LEGO NXT-G programming	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Line Follower	▪ Use light sensor in reflection mode in a lab setting ▪ Understand importance of calibration ▪ Differentiate raw and adjusted values ▪ Reinforce control flow logic ▪ Follow lab guidelines and roles	▪ Ability to comprehend effects of the environment on robot performance ▪ Ability to understand calibration and apply it	▪ Light sensor ▪ Calibration using the LEGO software ▪ Raw values versus adjusted values	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Photovore or Phototrope	▪ Use light sensor in ambient mode in a lab setting ▪ Distinguish reflected and ambient light ▪ Comprehend basic Boolean logic ▪ Play sounds on NXT	▪ Ability to comprehend and minimize effects an environment has on robot performance ▪ Ability to	▪ Reflected versus ambient light ▪ Performing math in NXT-G programming ▪ Logic (And, Or, Xor, Not) ▪ Comparison (greater than, less than,	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum

		▪ Follow lab guidelines and roles	compare and contrast	equals) ▪ Sounds on the NXT			
Third Quarter	Stopwatch	▪ Learn how NXT timers work ▪ Use NXT buttons ▪ Follow lab guidelines and roles	▪ Application and walkthrough of a more complex control flow	▪ Timer descriptions and functionalities ▪ Control flow with Stop blocks and loops with input logic	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Clapper	▪ Use sound sensor in a lab setting ▪ Understand robot inputs and outputs ▪ Know how to make general connections between blocks ▪ Understand how to use Variable blocks ▪ Follow lab guidelines and roles	▪ Grasp of changes in state during the history of a program ▪ Ability to understand different ways to communicate with a robot	▪ Sound sensor ▪ Variable blocks (creation, types, uses) ▪ Read versus write ▪ Ways to use connections between blocks	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Random Numbers	▪ Learn more complex control flow using switches with several cases ▪ Learn what a random generator is ▪ Follow lab guidelines and roles	▪ Critical thinking ▪ Analysis skills	▪ Range blocks ▪ All functions of a switch block, including flat view and default case ▪ Random blocks to generate random numbers	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Combine	▪ Understand the concept of encapsulation and its importance ▪ Learn about Calibrate blocks ▪ Follow lab guidelines and roles	▪ Ability to connect independent modules together ▪ Breakdown of a problem into smaller	▪ Calibration using blocks versus using programming environment menu ▪ Concept of encapsulation ▪ My Blocks in the NXT-G programming	▪ Lecture to explain how blocks work ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum

			independent tasks	environment			
Fourth Quarter	Text Messenger	▪ Introduce Bluetooth ▪ Understand asynchronous and NXT communication ▪ Send and receive messages between multiple NXT bricks	▪ Application of concept of Bluetooth technology	▪ Overview of Bluetooth technology and its applications ▪ Use Send Message and Receive Message blocks ▪ NXT mailbox system	▪ Lecture ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Differential Drive	▪ Understand mathematical foundation of differential steering ▪ Understand PID feedback control ▪ Implement and utilize proportional control on a robot ▪ Apply geometric concepts to robotics	▪ Estimation ▪ Calculating and planning ahead to solve problems	▪ Control with individual Motor blocks ▪ Relationship between distance traveled, wheel rotations, and wheel circumference ▪ Calculation of wheel speeds to achieve desired motion ▪ Smooth turning ▪ Overview of PID	▪ Lecture ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Obstacle Avoidance	▪ Understand the difference between reactive and predictive behavior	▪ Ability to use multiple sensors for the same task	▪ How a combination of sensors can solve a complex problem ▪ Reactive versus predictive behaviors ▪ Difference in detection and reaction to both near and far obstacles	▪ Lecture ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum
	Design and Create Own LEGO Robot	▪ Create a robot that can successfully solve a problem of the students choosing ▪ Apply good programming and	▪ Ability to narrow down a problem ▪ Critical analysis and decision on scope of a	▪ Review of good programming practices ▪ Review of good engineering design ▪ Brainstorming problems and	▪ Lecture ▪ Lab activity	▪ Projector ▪ Computer with LEGO software installed ▪ LEGO sets	▪ Hands-on practicum

		engineering practices	problem	ensuring appropriate level of complexity			

PROJECTS

First Quarter: Performance of all different Tribots
Second Quarter: A report on the compilation of all of the lab activities (either individually or in group)
Third Quarter: A report on the compilation of all of the lab activities (either individually or in group)
Fourth Quarter: Making own robot

REFERENCES

Power Point Presentations:

- Introduction to robotics
- Orientation for roles and lab activities
- Introduction to general sensors
- Basic mechanics

Websites:

- http://www.education.rec.ri.cmu.edu/roboticscurriculum/index_to_robotics.htm
- http://www.proactivepd.com.au/index.php?p=1_11
- http://www.owlnet.rice.edu/~elec201/Course_Notes.htm
- http://www.smartred.eq.edu.au/Science%20of%20Robotics/Main%20Page.html
- http://www.legoengineering.com/resources-a-downloads-mainmenu-60.html

Software:

- Lego Midstorms NXT v1.1

Role Guidelines

To ensure that all students participate, each will be assigned a number from 1 to 4 that will stay constant throughout the course. A system such as keeping track of the roles for each number in a spreadsheet can avoid many repetitions of a certain role assignment. The roles of engineer, programmer, project manager, and information specialist will rotate in each lab. There is an important rule to follow in keeping the roles restricted; on the day that number 1's are the project managers, for example, they cannot be doing the work of the engineers. A suggestion to enforce this is to penalize a group in which a student does the work of others. The following list details the tasks each role should perform:

ENGINEER

- Building the robot structure
- Placing sensors on the robot appropriately
- Knowing about mechanics
- Disassembling the robot at the end of class if necessary
- Ensuring working space is clean at the end of class

PROGRAMMER

- Programming the logic
- Downloading and running the program
- Testing and debugging
- Making sure all programs are saved on the computer and not just on the NXT

PROJECT MANAGER

- Reviewing the lab guidelines with the group each lab session
- Following all six steps (purpose, material and variables, process and logic, roles, Human Robot Game, and checkpoints)
- Making sure that the tasks get done in time to complete each checkpoint successfully
- Scheduling and managing team members
- Coordinating team members' implementations of the design to fulfill the goal

INFORMATION SPECIALIST

- Documenting all the steps and notes in the group's notebook
- Helping the engineer and programmer if they have problems
- Delivering the group's presentation at the end of each session
- Keeping track of all parts used and ensuring none are missing by the end of class
- Taking notes on the process of reviewing lab guidelines with the group
- Updating the group on work performed last time when continuing lab sessions
- Drawing the Robot Layout Diagram and Logic Flow Chart and describing each component of them

Lab Guidelines

During initial classes that introduce students to hands-on activities with the LEGO sets, the teacher should go through this model with the entire class. After the students acquire proficiency, the project manager should be in charge of reviewing this model every time with their own groups.

PURPOSE

- Explain the robot to be built and task to be accomplished

MATERIAL AND VARIABLES

- Delineate essential parts that will be used in the robot
- Define what sensors and actuators will be used
- Define variables to be controlled in logic
- Determine inputs and outputs
- Fit the above into the bigger picture of the Robot Layout Diagram

During each class, groups should review the Robot Layout Diagram to determine how it solves the purpose, where the materials and variables fit in each step, and how the logic, sensors, inputs, and outputs tie in. This discussion would usually be lead by the project manager. It is essential that the information specialist takes notes on this completed diagram so the teacher can check it before letting the group continue with the robot's implementation.

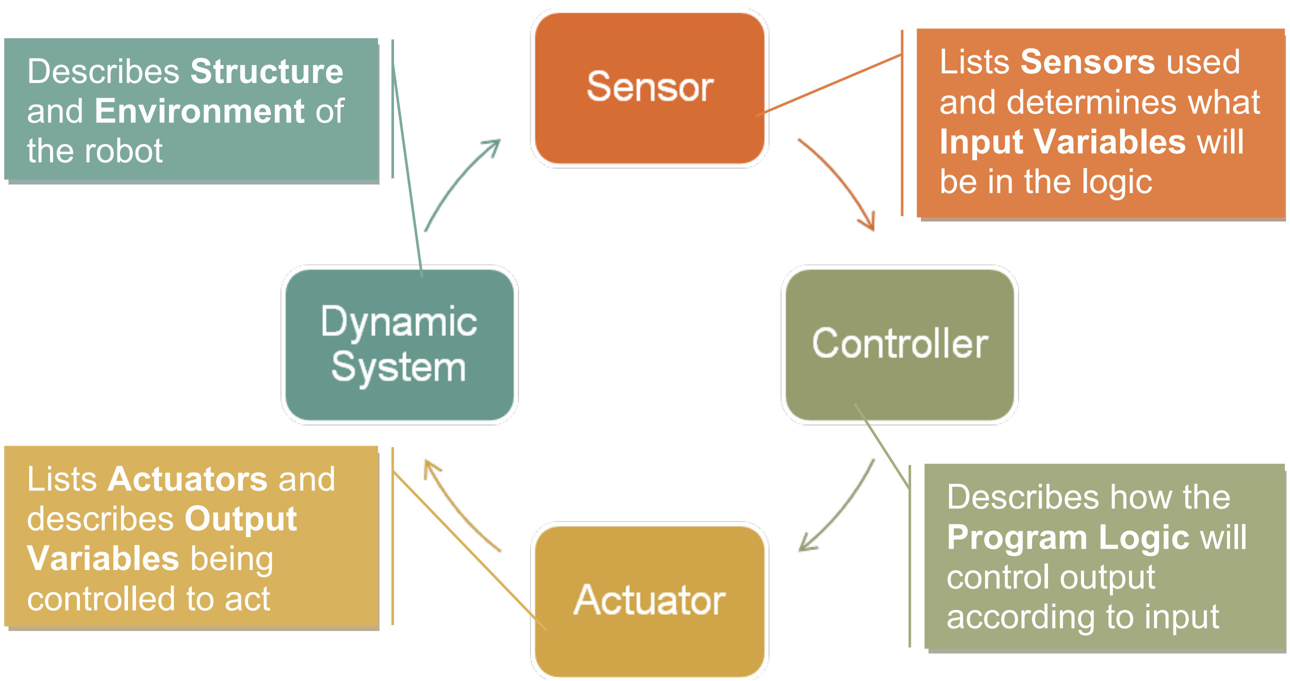


FIGURE 4: ROBOT LAYOUT DIAGRAM

PROCESS AND LOGIC

- Review with students and attempt to simplify logic into Boolean or numerical values
- Discover with students the relationships between variables defined in the previous step and the logic to perform desired actions
- Draw the Logic Flow Chart that represents the logic to be programmed

ASSIGN ROLES

- Split roles for students in each group according to numbering system

HUMAN ROBOT GAME

- A "blinded" student is a robot
- A student facing away from the robot is the programmer
- Other students act as scenery or extra parts of the robot according to tasks

For more information, refer to Appendix F.

CHECKPOINTS

- Explain to class that the teacher will come and check the progress of robots at these intervals of time
- Project manager should make sure that the group is on time for each of these checkpoints, which will occur over multiple class sessions and must be done in order:
    a. Complete all lab guideline steps (especially Robot Layout Diagram and Logic Flow Chart)
    b. Human Robot Game
    c. Build*
    d. Program*
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

* *Happens concurrently*

NXT-G Programming Sequence for Labs

During the first lab session, the teacher should introduce the basic administrative navigation of the NXT-G programming editor:

- Copy, paste, and undo
- Moving blocks around
- Navigating through the editor
- Creating and opening profiles
- Saving and opening files
- Checking NXT brick status
- Downloading programs
- Differentiating inputs and outputs (introduced as necessary with different blocks)
- Connecting data wires

The following table delineates the types of block and functions that should be introduced before students follow the lab guideline procedure. Teachers should provide clear explanations of how the blocks work with some examples so students can apply them as necessary in the lab exercisess.

TABLE 5: NXT-G PROGRAMMING SEQUENCE

Lab	Block	Functions
Tribot base	Move	▪ Ports ▪ Direction: forward, backwards ▪ Power ▪ Duration: rotations
Tribot bumper	Move	▪ Duration: unlimited
	Wait	▪ Until (explain how it differs from wait while) ▪ Control: touch sensor
	Touch Sensor	▪ Port ▪ Action: pressed, released, bumped
Tribot grabber	Move	▪ Direction: stop ▪ Port ▪ Duration: seconds ▪ Power
Tribot sound sensor	Wait	▪ Control: sound sensor
	Sound Sensor	▪ Port ▪ Compare: sound level, logic
	Calibrate sensors	▪ Use menu option, not Calibrate block
Tribot light sensor	Move	▪ Duration: degrees
	Wait	▪ Control: light sensor
	Light Sensor	▪ Port

		▪ Compare: light intensity, logic ▪ Function: ambient versus reflected
Tribot ultrasonic sensor	Wait Ultrasonic Sensor	▪ Control: ultrasonic sensor ▪ Port ▪ Compare: distance ▪ Show: inches versus centimeters
	Inputs, outputs	▪ Explain difference and how they look like in the blocks
Bumper Robot	Move	▪ Steering ▪ Ways to use connections
	Rotation Sensor	▪ Port ▪ Action: reset, read ▪ Compare ▪ Inputs and outputs ▪ Talk about accuracy of measurement (22 degrees)
Object Follower	Loop	▪ Control: forever
	Switch	▪ Control: sensor
	Display	▪ Image ▪ Text
	Text	▪ Concatenation ▪ Inputs and outputs
	Number to Text	▪ Inputs and outputs
	Ultrasonic Sensor	▪ Input: trigger point ▪ Output: yes/no, distance
Line Follower	*Raw versus adjusted values*	
	Light Sensor	▪ Input: trigger point, generate light ▪ Output: yes/no, intensity, raw value
Photovore and Phototrope	Logic	▪ And, Or, Xor, Not ▪ True versus False
	Compare	▪ Greater Than, Less Than, Equals
	Math	▪ Addition, subtraction, multiplication, division ▪ Inputs and outputs
	Switch	▪ Input: number, text, logic
	Sound	▪ Action: sound file, tone ▪ Control: play, stop ▪ Volume ▪ Function: repeat ▪ Wait for completion
Stopwatch	Display	▪ Review how to display text
	NXT buttons	▪ Similarity to touch sensor ▪ Button: enter, left, right
	Loop	▪ “Until” sensor ▪ Input: logic (switch blocks are not allowed in this lab)
	Wait	▪ Control: time, sensor
	Stop	▪ Input: Yes/No
	Timer	▪ Explain how they run (since the start of all programs)

		▪ Timer numbers ▪ Action: reset, read ▪ Output: time, yes/no
Clapper	*General sensor inputs and outputs*	▪ Reinforce trigger point, greater/less, yes/no
	Sound Sensor	▪ Output: sound level
	Move	▪ Input: duration (explain exceptional behavior: if set to rotation, it really uses degrees, and if set to seconds, it really uses milliseconds) ▪ Ways to use connections
	Variables	▪ How to create new variables ▪ Actions: read, write ▪ Type: logic, number, text
Random Numbers	Range	▪ Operation: inside, outside ▪ Explanation of endpoints being inclusive for inside range, and exclusive for outside range ▪ Inputs and outputs
	Random	▪ Range (0 to 32767 when entered manually, even though the slider only goes to 100) ▪ Explanation of range endpoints being inclusive ▪ Inputs and outputs
	Switch	▪ Control: value ▪ Type: number ▪ Display: flat view ▪ Defining custom conditions
	Loop	▪ Count ▪ Time
Combine	Calibrate	▪ Contrast with menu calibration ▪ Need separate blocks for setting minimum and maximum values ▪ Action: delete, calibrate
	NXT Buttons	▪ Action: pressed, released, bumped
	My Blocks	▪ Basic ▪ How to create inputs and outputs
Text Messenger	Send Message	▪ Connection ▪ Message: text, logic, number ▪ Mailbox ▪ Inputs
	Receive Message	▪ Message: text, logic, number ▪ Mailbox ▪ Inputs and outputs
Differential Drive	Motor	▪ Direction ▪ Control: power ▪ Duration ▪ Wait for completion

Recommended Team Structure

GROUPS OF 5 OR FEWER STUDENTS

Teams of five or fewer students work together to plan, build, and program a robot in order to do a specific task. Each student will be assigned the role of engineer, programmer, project manager, or information specialist (for detailed role descriptions, see the Role Guidelines section). Each of these roles will have different responsibilities; however, all roles will go through the design process, work together, and help each other out when needed in order to create a functional robot.

When there are more than four students, the additional one can also be assigned the role of engineer or programmer as an additional helper.

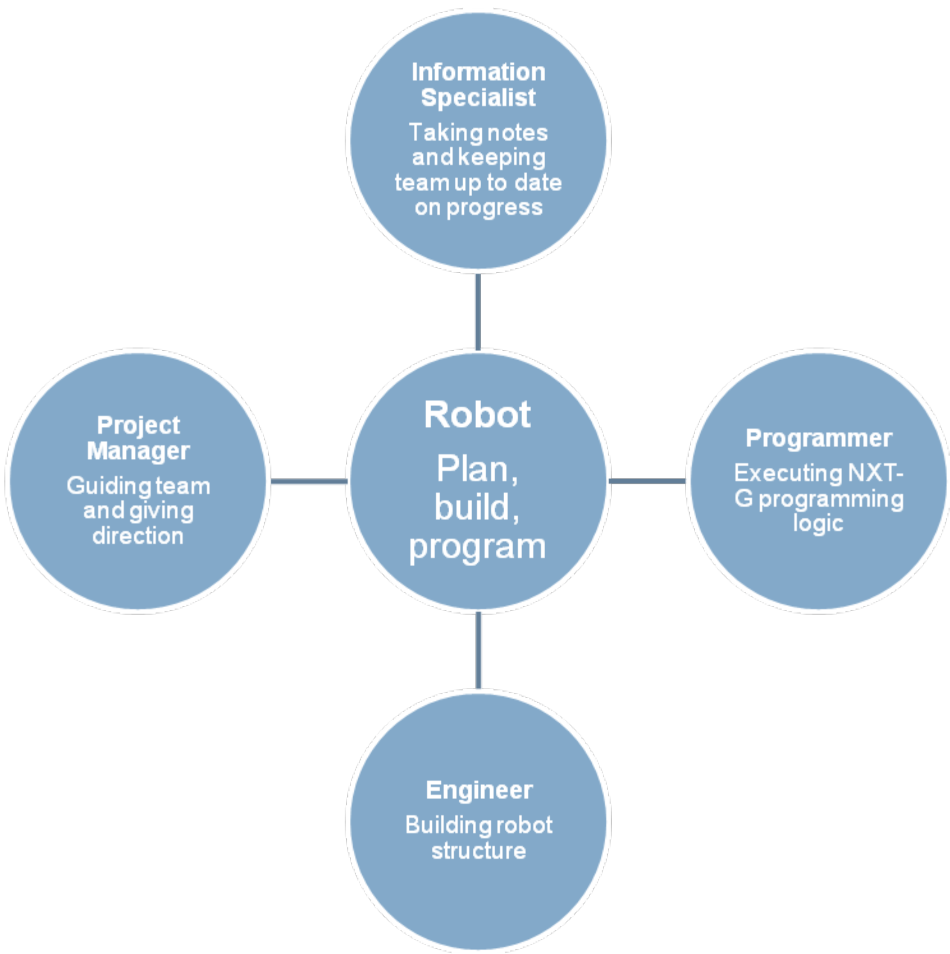


FIGURE 5: TEAM STRUCTURE FOR 5 OR FEWER STUDENTS

Best for:

- Small student-to-kit ratio (for instance 1 kit for a team of 4 students)

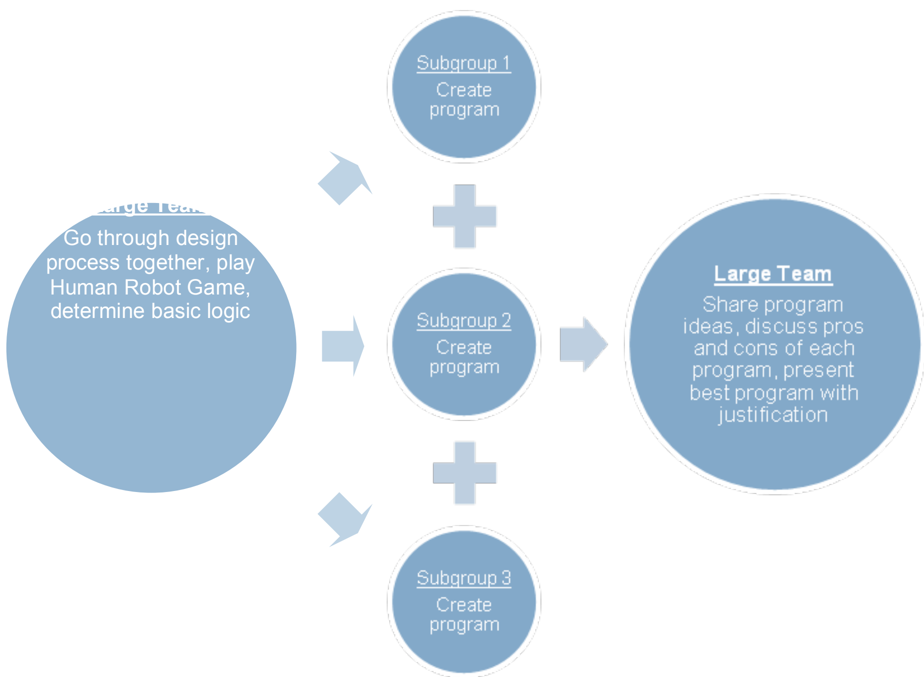
- Small projects and non-complex programming so that the other students who do not have the Programmer role will not be idle and will remain engaged
- Assigning roles to all students to motivate them to actively participate in teams

GROUPS OF 6 OR MORE STUDENTS

The large teams will be composed for 3 subgroups of 2 students each (6 students total). Different subgroups will alternate in leading the large team in the planning phase of each project. The leading pair of students will go through the design process with one of those two students taking on the role of Project Manager, while the other will take the role of the Information Specialist.

Once the large team is finished with the design and planning phase of the project, the subgroups will proceed to programming the logic individually as pairs. As the time spent using the physical robot is a minimal part of the entire process, the subgroups can take turns testing their programs on the large team's robot during the programming phase.

When the programming phase is complete, the subgroups will gather again as a large team. This time, each subgroup will explain their individual program logic to the entire large team. Then the large team will decide which program is the best and list the reasons for making that choice. This best design will be presented to the entire class during the presentation and demonstration checkpoint.



Best for:

- Large teams
- Maximizing the number of students actively programming during one particular time
- Programming-intensive robots

- Low budgets

Recommended Milestone Curricula

In order for OLLCF to establish a robotics institute and offer a robotics minor or major, there are several recommended milestone steps to accomplish. It must be emphasized that all criteria in a phase should be met before proceeding to the next one, as a solid foundation and mastery of each skill is essential for the success of the program. We stress that this is only a guideline and later phases should be adjusted to fit OLLCF's future situation as it progresses. Furthermore, we strongly advise seeking additional recommendations and guidance from faculty members and experts at other institutions in the field of robotics as this timeline is prepared by a group of undergraduates with limited backgrounds themselves.

PHASE I

Offer an introductory robotics class at the high school and college level. This course would serve the purpose of fostering interest in the field, giving a broad foundation of knowledge, teaching the fundamental basics of robotics, and establishing a realistic vision of what is needed to build a robot.

In addition, courses offering a solid foundation in general programming logic and the C language are essential for later higher level classes.

PHASE II

Establish the College of Engineering to compose minimally of mechanical engineering and electronics and communication engineering. Critical topics useful for robotics are listed below according to their major discipline along with general knowledge that all students should obtain. These are the necessary basic courses that serve as a foundation and prerequisites for later courses.

Mechanical Engineering

Curriculum should include:

- Physics (mechanical): Newton's law, momentum, force, equilibrium, free body diagrams, work, energy, velocity, acceleration,
- 3-D modeling of designs (programs similar to AutoCAD or SolidWorks should be used): basic modeling of components, drafting

Possible lab activities include:

- Measuring the torque of a motor
- Building a truss bridge out of balsa wood to support the maximum weight
- Building a mousetrap car to go the maximum distance forward

Laboratory equipment:

- Materials to construct lab activity (e.g., wood for building bridge, mouse trap for car)
- Pliers
- Clamps
- Screw drivers
- Drills

- Files
- Scissors
- Glue
- Space for large tables

Electronics and Communication Engineering

Curriculum should include:

- Physics (electromagnetism): electricity, electrostatics, electric fields, Gauss's law, electric potential, simple circuits, magnetism, magnetic force, magnetic fields, induction
- Basic circuits: electrical components (resistors, diodes, transistors, capacitors), Kirchhoff's laws, Ohm's law, current, voltage, operational amplifiers, system decomposition

Possible lab activities include:

- Measuring voltage, resistance, and current in simple circuits
- Digital circuit simulation

Laboratory equipment:

- Breadboards
- Multimeters
- Electrical circuit components (resistors, capacitors, etc.)
- Power supplies

General Knowledge

In order to take third or fourth year classes, a strong foundation in these topics is required. It is assumed that basic algebra and trigonometry is already known.

Curriculum should include:

- Differential calculus: functions (inverse, exponential, logarithmic, hyperbolic), limits, derivatives, curve sketching, minimum and maximum values
- Integral calculus: definite and indefinite integrals, application of integration, methods of integration (substitution, parts, trigonometric, partial fractions)
- Basic programming in C: data types, operators, operands, control flow, functions, conditional statements, loops, arrays, pointers, lists

Optionally, OLLCF can offer summer robotics workshops for local students using LEGO Mindstorms NXT sets to stimulate interest in the field among the community. Partnering with the Pinoy Robotic Games Foundation[3] would facilitate this as an option.

PHASE III

[3] http://www.pinoyrobotgames.org/

Establish more advanced topics to build upon the classes introduced in Phase II. Instructors should have thorough knowledge in these topics beforehand, having taken them during their college studies. These courses should be taught after the Phase II classes are taken by students as they complete the general knowledge that all engineering and computer science concentrations rely on. These courses will later serve as a student's foundation for expanding into different specializations, including robotics. See Appendix G for recommended textbooks and online resources applicable to these classes.

Mechanical Engineering

- Force analysis: rigid body statics, tension, compression, torsion, shear, bending, deflections
- Stress analysis: deformation, combined loading and stresses, 2-D elasticity, failure modes, properties for common materials
- Dynamic systems: rigid body reactions, moments, kinematics, relative motion, momentum, different coordinate systems (polar, rectangular, intrinsic)

Electronics and Communication Engineering

- Analog circuits: circuit analysis, passive and active components, energy storage elements, first- and second-order systems, sinusoidal steady-state response
- Digital circuits: flip-flops, digital number systems, computer architecture, sequential logic design and optimization, Boolean algebra, basic processor organization and instruction sets, hardware description languages (e.g., Verilog, VHDL)
- Signals: continuous and discrete time signals, frequency domain, impulse response, convolution, sampling, Fourier transform, mathematical analysis tools such as MATLAB

Equipment to add to the electronics and communication engineering laboratory includes soldering irons, oscilloscopes, and function generators.

Computer Science

- Data structures: stacks, queues, binary trees, balanced trees (e.g., AVL, red-black, B, 2-3-4, splay), hash tables
- Advanced C programming: dynamic memory allocation, machine-level code, performance evaluation and optimization, memory organization, concurrency
- Basic algorithms: general analysis, recursion, breadth-first and depth-first search, searching algorithms, comparison of sorting algorithms (e.g., bubble, shell, selection, merge, quicksort)

General Knowledge

- Multi-variable calculus: vectors, coordinate systems (polar, cylindrical, spherical), multiple integrals, line integrals, surface integrals, divergence theorem
- Differential equations: ordinary differential equations, first- and second-order differential equations, Laplace transforms, Fourier series
- Complex analysis: rectangular and polar representations, complex exponential, Euler's formula, complex powers, complex roots, complex logarithms, complex differentiation, analytic functions, Cauchy-Riemann equations
- Linear algebra: vectors, matrices, vector spaces, real and complex eigenvalues and eigenvectors, linear transformations
- Probability and statistics: conditional probability, independence, random variables and expected values, random processes, distribution functions, law of large numbers, central

limit theorem, confidence intervals, maximum likelihood estimation, linear regression, analysis of variance

PHASE IV

Establish an advanced robotics class that would program LEGO Mindstorms NXT robots using C (for example, in creating a maze solver).[4] At this point, we recommend that there be at least two instructors well-versed in each topic that is taught so as to avoid overloading any one faculty member and ensure the quality of education remains consistent for all students.

In this phase, OLLCF can set up technical and vocation programs to train students as technicians. Similar to obtaining an associate's degree, students in these vocational programs would also need the foundational courses detailed in Phase II. For the courses in Phase III, however, vocational students would only take some of them according to their pursued specializations. A robotics technology program can be set up in conjunction with electronics, telecommunications, and mechanic programs. We strongly recommended contacting other colleges and universities that offer similar technical trades for proper guidance.[5]

At this point, the high school can optionally submit a robot into the FIRST Robotics Competition, International Robotics Olympiad, or similar competitions with the guidance of the robotics faculty.[6]

PHASE V

Train staff in advance robotics topics in order to equip them with the knowledge necessary for teaching future classes. Full understanding of the material and expertise in one's area is essential before proceeding to teach it to others. As these topics may not be offered at local universities, the training of instructors in these areas may require ample time for self-teaching by reading textbooks, journals, and research papers. Faculty members teaching these classes should have doctorate-level education, although not necessarily in the specific specializations, as such individuals have extensive experience when it comes to demanding research and independent study. These courses are applicable to those students from different disciplines wanting to specialize in robotics.

Recommended essential topics:

- Control systems: control of robotics mechanisms, systems theory, control of dynamic systems, motion planning, collision avoidance, adaptive control, hybrid control
- Robotic manipulation: computational models of objects and motions, mechanisms of robotic manipulators, planning and programming of robotic actions
- Kinematics and dynamics: linkage systems, reciprocating engines, industrial machinery, computer analysis, synthesis of mechanisms
- Mobile robots: control, localization, mapping, perception, planning algorithms, communication between multiple mobile robots

---

[4] An example of such a class using advanced programming of LEGO Mindstorms NXT robots: http://www.generalrobotics.org/

[5] A list of colleges in the United States that offer a robotics technician degree: http://www.ed-reference.us/00110/technical/robotics-technology,-technician/colleges

[6] http://www.usfirst.org/ and http://www.iroc.org/

- 3-D model analysis and simulation (mechanical engineering students only): programs similar to CosmoWorks
- Embedded systems (electronics and communications engineering students only): embedded architectures, interaction with devices, concurrency, real-time principles, real-time resource and device management
- Advanced algorithm analysis (computer science students only): graph algorithms, probabilistic algorithms, dynamic programming, approximation algorithms, complexity theory

Note that many of these classes (for example, 3-D model analysis and simulation) require high-performance computers that would have to be purchased beforehand.

PHASE VI

Faculty and students will be able to design, build, and program their own basic robotics systems. Laboratory equipment from earlier mechanical engineering and electronics and communication engineering classes can be used for constructing these robots. Senior-level design classes may be included wherein students complete an entire robot of their own design in teams.

PHASE VII

Instructors learn and teach more advanced robotics topics that would offer concentration and expertise in the subfields of robotics. The depth of these subjects usually requires instructors to have extensive research experience with a PhD thesis in the given area. It is recommended that each instructor concentrate on a different topic to diversify the knowledge pool and to maximize the variety of courses that can be taught.

Advanced topics and concentrations include:

- Computer vision: machine vision, sensing and perception, 2-D image analysis, physics-based vision
- Mechatronic design: integration of mechanisms, electronics, and computer control
- Artificial intelligence: cognitive modeling, machine learning, data mining, decision-making machines, adaptive algorithms

Given the advanced research involved in these areas, requesting government sponsorships would be particularly helpful, as students and faculty can learn the subject matter while working on applicable government projects.

PHASE VIII

Establish a research institute with an advanced laboratory used to build robots. At this point, OLLCF can gather research grants to build robots to accomplish specific tasks. In addition, students and faculty can build complex robots for international competitions and submit papers to research journals and conferences.

Step 5: Propose Laboratory Setup

With our recommendation to use the LEGO Mindstorms NXT sets, there would be no need for an exclusive space dedicated to the introductory robotics classes, nor any extra costs for specialized robotics equipment.

The implementation of our proposed classes would only require modifying the current setup of a computer lab by placing larger tables in the room to serve as workspaces for building and testing robots. The current high school computer lab is one possible space that can be used for this, as it already has one large table as well as enough computers for pairs of students to develop programs for the LEGO Mindstorms NXT robots. In addition, the computer lab is one of the few rooms with air conditioning installed; this would provide an adequate temperature for the operation and maintenance of the robots. In most class sessions, students will perform hands-on building, programming, testing, and debugging of robots, so having a computer lab as the location for regular class meetings is ideal.

The following list details the adjustments that should be made to the computer lab to accommodate the robotics classes:

- Install the NXT-G software on each computer
- Add additional tables to provide for each team to build and test its robot

In addition, there are items that should be purchased for administrative purposes of the kits. Given that each LEGO set consists of more than 400 pieces, we recommend using storage bins with separate compartments that can categorize different pieces. Purchasing such containers can protect sensitive equipment like sensors and motors and minimize the loss of small pieces.

ABOUT THE CONSULTANTS

The student consultants are available for further guidance and recommendations as OLLCF moves forward with its robotics program. Their e-mail addresses are listed below along with brief biographies for each one.

Steven Elia

selia@alumni.cmu.edu

Steven graduated from Carnegie Mellon University in May, 2009 with a Bachelor of Science in Electrical and Computer Engineering with an Additional Major in Computer Science. Interested in operating system development and the field of software theory, he will begin his pursuit of a Master of Science in Computer Science at Stanford University starting in September, 2009. Robotics has been a long-time hobby of his since high school, and he looks forward to continuing software development for robotic systems as a side interest, including his own version of a Rubik's cube solver.

Siying Diana Hu

sdhu@andrew.cmu.edu or siyinghz@gmail.com

Diana will graduate from Carnegie Mellon University with a Bachelor of Science in Electrical and Computer Engineering with minors in Robotics and Music Technology in May, 2011. She will finish her degree and prepare to study for a masteral or doctoral degree in robotics or computer engineering. She is a member of the Robotics club at Carnegie Mellon. Although she is a Chinese descendant, she was born and grew up in Chile. She has worked in some local volunteering programs in Chile.

Her interests in global development include developing sustainable technological communities and information and communication technologies for development. Aside from this, she plays the violin and is looking into the field of music technology.

Sixiao Joy Liu

sixiao.liu@gmail.com

Joy graduated from Carnegie Mellon University with a Bachelor of Science in Mechanical Engineering in May, 2009. After the completion of the Technology Consulting in the Global Community program in the Philippines, she will begin her career at Intel as a process engineer. Her interest in manufacturing will lead her to pursue a Master of Business Administration in the future. She hopes to one day help bring innovative products onto the market in order to uplift the standard of living for different people and touch their lives in a positive way.

APPENDIX A: OLLCF ROBOTICS COURSE SYLLABUS

I. COURSE NO

CS13

II. COURSE TITLE

Modeling Simulation

III. COURSE DESCRIPTION

This course covers discrete event system modeling and simulation, as well as selected topics in the general field of computer simulation. It also focuses on the use of Robo-box for robotics development. Topics include microcontrollers, sensors, structural components, computers, motors, wheels, and remote control.

IV. GENERAL OBJECTIVES

1. Present concepts of Modeling and Simulation applicable to a wide variety of systems
2. Provide hands-on experience with Modeling and Simulation

V. PRE-REQUISITE:

CS/2/CS5

VI. EXIT REQUIREMENTS

Robotics

VII. TERMS

3 hours per week

1 semester

VIII. CREDITS

3 Units

IX. COURSE CONTENT

No. of hours	Contents	Instruments, Materials, and Methods	Evaluation
6 hours / 2 weeks	▪ Introduction to Modeling and Simulation ▪ General overview of Modeling and Simulation ▪ System concepts ▪ Decomposition and composition ▪ Levels of systems knowledge	▪ Lecture ▪ Laboratory Demonstration ▪ Hands-on	▪ Recitation ▪ Quizzes ▪ Assignments ▪ Research ▪ Projects ▪ Examinations
15 hours / 5 weeks	▪ Robo-box overview ▪ Microcontroller ▪ Sensors ▪ Structural components ▪ Computer ▪ Motors and wheels ▪ Remote Control	▪ Lecture ▪ Laboratory Demonstration ▪ Hands-on	▪ Recitation ▪ Quizzes ▪ Assignments ▪ Research ▪ Projects ▪ Examinations
15 hours / 5 weeks	▪ Accessories ▪ Touch / switch sensor ▪ Light sensor / detector ▪ Light reflector ▪ LDR ▪ GP2D120 sensor ▪ Wheels and DC motors ▪ Tracks and sprockets ▪ RF modules	▪ Lecture ▪ Laboratory Demonstration ▪ Hands-on	▪ Recitation ▪ Quizzes ▪ Assignments ▪ Research ▪ Projects ▪ Examinations
18 hours / 6 weeks	▪ Building a Basic Robot ▪ Construction phase ▪ Programming the Basic Robot ▪ Simple movement for Model I Basic ▪ Building the Bumper Robot ▪ Sub-functions ▪ Building the Light Robot ▪ Building the Surveyor	▪ Lecture ▪ Laboratory Demonstration ▪ Hands-on	▪ Recitation ▪ Quizzes ▪ Assignments ▪ Research ▪ Projects ▪ Examinations

	▪ Building the Sumobot		

X. REFERENCES

- http://www.inexglobal.com

Prepared by:

Gerry P. Lopez, MCS

Noted by:

Cresencia A. Rasco, PhD

VP for Academic Affairs

APPENDIX B: WORKSHOP SYLLABUS

In order to ensure the sustainability of the robotics curriculum, this workshop aims to offer the attendees a detailed understanding of the material to be taught in the two introductory classes to be opened at the high school and college levels. In addition, the classes will be structured in a way as to adequately prepare the attendees to write their own lesson plans to fit each of their teaching styles. To accomplish this, the first portion of each class will be tailored towards instruction of robotics material, while the second portion will be an open-forum discussion regarding how to teach the material to students, such as the examples to use, demonstrations to hold, and exercises to perform. During the brainstorming discussion forum, all attendees will be asked to actively participate for the best result.

The introductory robotics class will be taught over the course of one year at the high school level and one semester at the college level. However, because of the time constraint on the program, this workshop must proceed at an accelerated rate and will last five weeks. It is requested that the attendees allot a minimum of 6.5 hours each week for the workshop learning session and 2.5 hours each week for the discussion forum. Tests will be administered as necessary during the five week period to provide feedback. Furthermore, additional practice will be offered to reinforce the learning and provide a chance for individually-tailored feedback to be given to the attendees. Office hours will also be open for anyone who wants additional instruction, feedback, or general input regarding the curriculum. Due to scheduling conflict, those who will miss workshop sessions should come to office hours to remediate material covered during that period.

Resources Needed for Classes

- 1 room to accommodate around 10 people
- 1 LEGO Mindstorms NXT set
- 1 projector
- 1 laptop

Additional Resources

- 1 Computer (for additional assignments)

Attendee Requirements

- No prerequisite knowledge necessary
- Prior knowledge of programming and electronic components preferred
- General interest in math and science

Schedule of Classes

Location: High School AVR

MWF: 8:00 am – 10:00 am

TTH: 9:30 am – 11:00 am

Office Hours

Location: Internet Room

M – F: 11:00 am – 3:00 pm

Course Outline

TABLE 6: TENTATIVE WORKSHOP SCHEDULE

	Week 1	Week 2	Week 3	Week 4	Week 5
Mon	Introduction to Robotics	Introduction to LEGO	Build Object Follower	Third Party Programming Languages	Finch
Tue	Sensors and Actuators	Introduction to Programming with LEGO	Advanced Robotics Introduction and Signals	Motion and Movement	Cognitive Robotics and Artificial Intelligence
Wed	Basic Electronics	LEGO Sensors and Actuators	Feedback Controls	Kinematics	TBD
Thur	Basic Mechanics	My First Robot	Microcontrollers	Path Panning	TBD
Fri	Simple Machines	Build Line Follower	Build Obstacle Avoidance Robot	Vision Detection	TBD

TABLE 7: ACTUAL ACCOMPLISHED WORKSHOP SCHEDULE

	Week 1	Week 2	Week 3	Week 4	Week 5
Mon	Introduction to Robotics	Introduction to LEGO and My First Robot	Build Object Follower	Clapper	Differential Drive
Tue	Sensors and Actuators	Introduction to Programming with LEGO	Introduction to Lab Guidelines and Roles	Stopwatch	Obstacle Avoidance
Wed	Basic Electronics	LEGO Sensors and Actuators	Clapper	Random Numbers	Obstacle Avoidance
Thur	Basic Mechanics	Build Bumper Robot	Clapper	Combine	Text Messenger
Fri	Simple Machines	Build Line Follower	Photovore (individual test)	Differential Drive	Create Own Robot

APPENDIX C: RECOMMENDED HIGH SCHOOL TIMELINE

The introductory robotics course for the high school will be available during the fourth year under the computer studies subject sequence. Our presented curriculum is intended to be taught for an entire year at the high school level. Presented below is our suggested sequence for high school computer classes that will ensure robotics can fit into the fourth year with enough time to teach everything.

First Year

I. Basic Computing Use
II. OpenOffice.org Text Document and Presentation
III. Open Office Spreadsheet
IV. Introduction and Advanced GIMP

Second Year

I. Advanced GIMP and GIMP Animation
II. Introduction 3D Blender
III. Advanced 3D Blender
IV. 3D Blender Animation

Third Year

I. HTML and Introduction to Programming
II. Introduction to Java
III. Intermediate Java
IV. Introduction to C

Fourth Year

I. Robotics 1
II. Robotics 2
III. Robotics 3
IV. Robotics 4

APPENDIX D: SOLUTIONS TO LAB ACTIVITIES

Bumper Robot

This lab must use the Rotation Sensor block.

1) **Purpose:** Make a robot that drives straight until it bumps into an object, at which point it turns 180 degrees and drives back at twice the original power. It stops at the same point it started.
2) **Material and Variables:** Touch sensor, rotation sensors, degrees (input), bumping Yes/No (input), steering(output), power(output), duration (output)

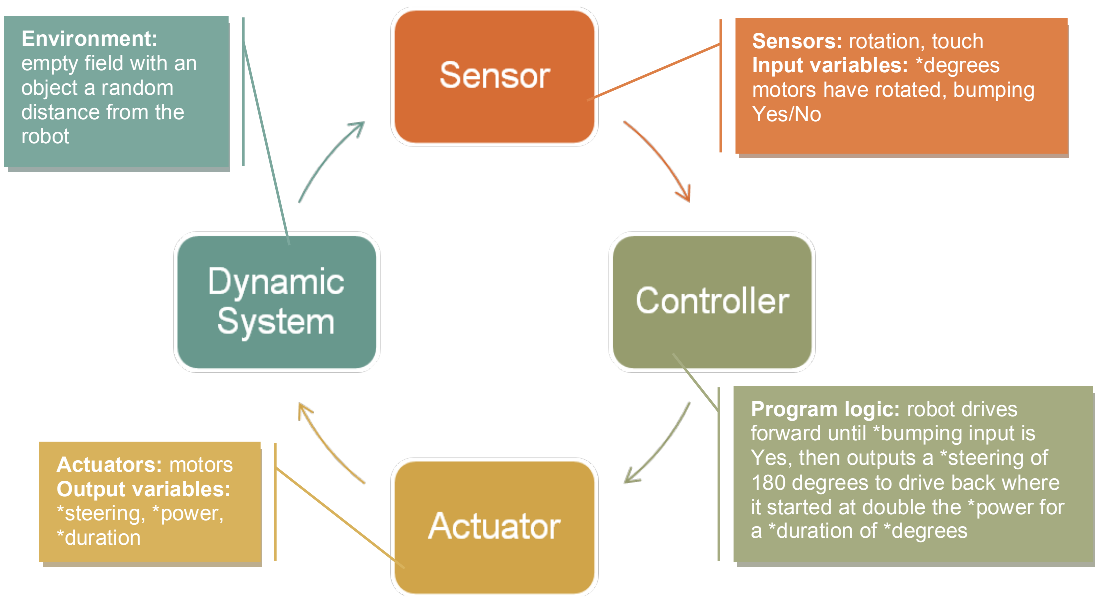


FIGURE 7: ROBOT LAYOUT DIAGRAM FOR BUMBER ROBOT

3) **Process Logic:**

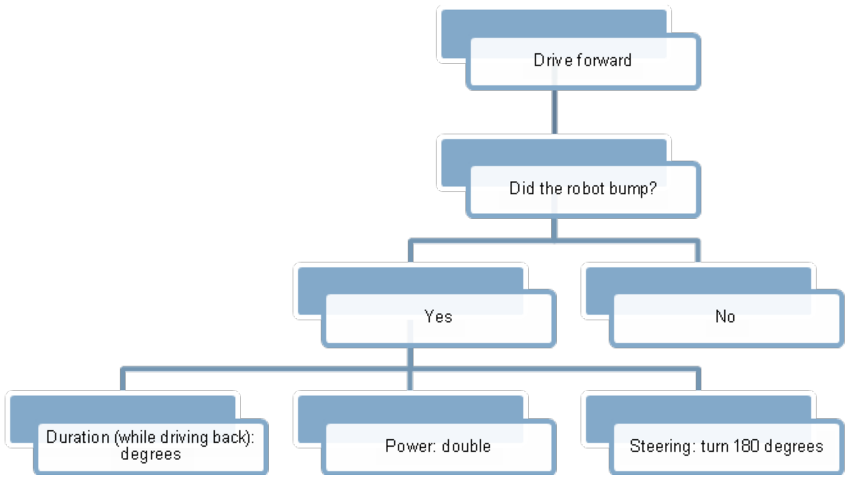


FIGURE 8: LOGIC FLOW CHART FOR BUMPER ROBOT

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the stationary object, another is the programmer, and the last student is the robot. The object stands a random distance directly in front of the robot while the robot waits for commands. The programmer, who cannot see what the others are doing, commands the robot take one step forward, then sense if it has bumped the object. The robot responds with “not pressed,” and the programmer asks the robot move forward and then sense again. This continues until the robot comes in contact with the object and reports a “yes pressed.” The programmer then asks how many steps the robot has taken and then commands the robot to turn around and take that many steps back.
6) **Checkpoints:**
   a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
   b. Human Robot Game
   c. Build
   d. Program:
      i. Have the robot make a 180 degrees turn
      ii. Drive forward until the robot bumps the object, then make a 180 degrees turn
      iii. Perform (ii) and then drive back to the start based on the number of motor degree rotations
   e. Test and debug
   f. Present and demonstrate robot to class or teacher

Additional activities:

   a. Instead of making a 180 degrees turn after it bumps, make a 90 degrees turn to the left and drive for 3 seconds. Then make another 90 degrees left turn and drive according to the number of rotations from the first drive forward before bumping.

Object Follower

1) **Purpose**: Make a robot that follows an object and continuously displays on the screen “_ cm from object” where _ is the distance. It should keep a constant 10 cm distance from the object.
2) **Materials and Variables:** Ultrasonic sensor, distance (input), power (output), direction (output), display (output)

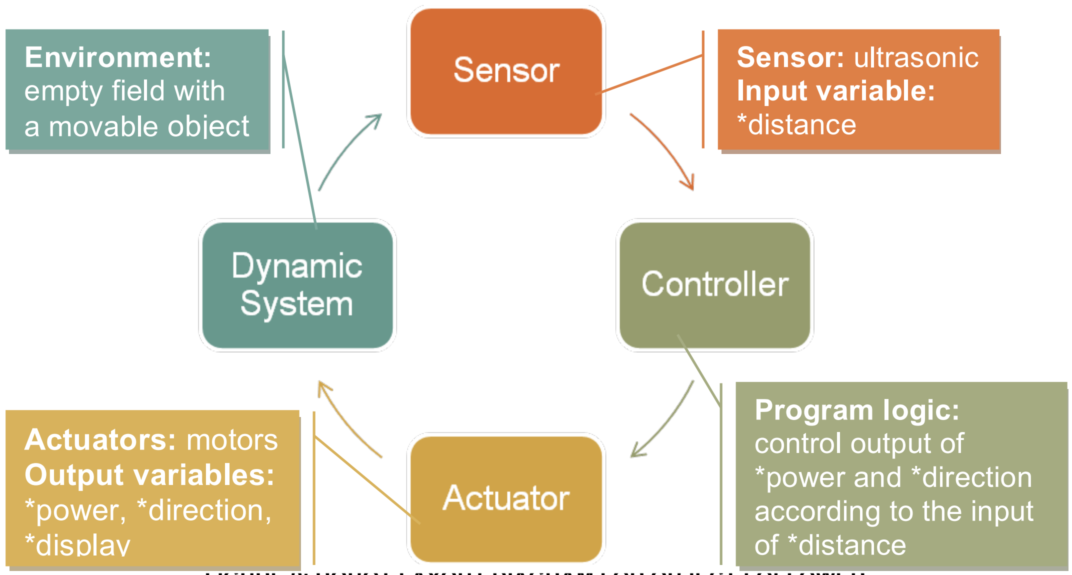


FIGURE 9: ROBOT LAYOUT DIAGRAM FOR OBJECT FOLLOWER

3) **Process Logic:**

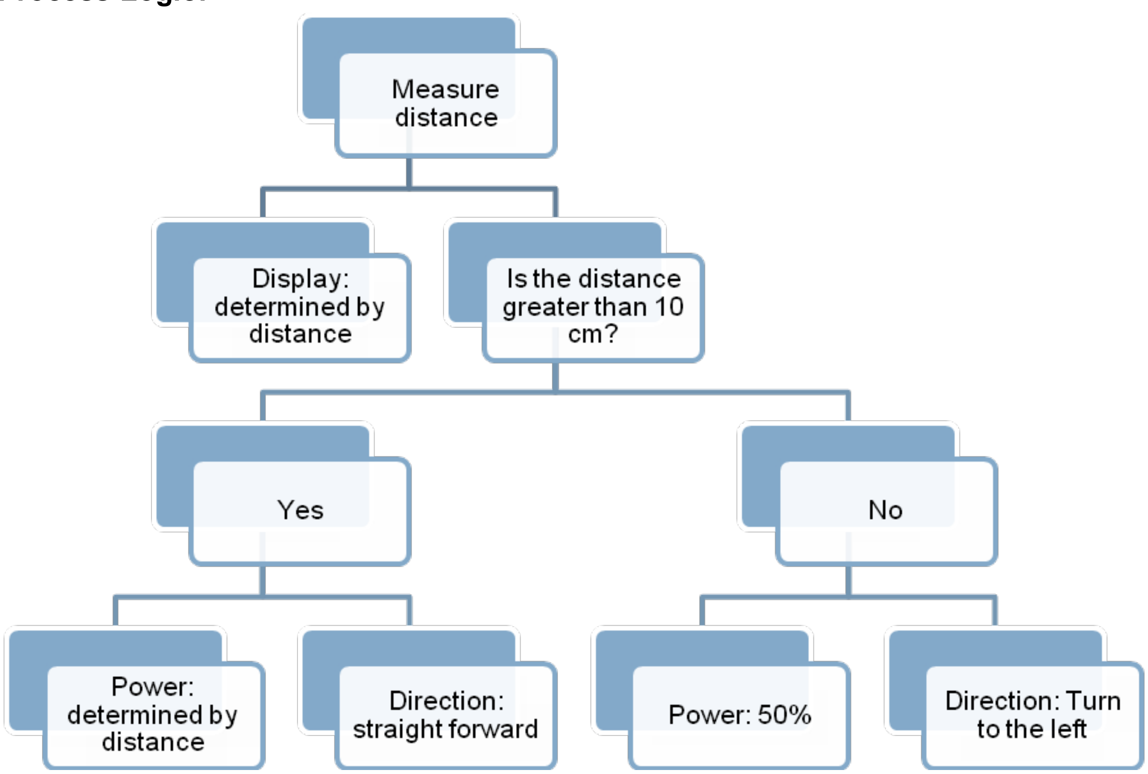


FIGURE 10: LOGIC FLOW CHART FOR OBJECT FOLLOWER

4) **Assign Roles**
5) **Human Robot Game:** To simulate the task, one student acts like a movable object, another is the programmer, and the last student, who should be “blind,” is the robot. The object remains still in a random position while the robot waits for commands. The programmer, who cannot see what the others are doing, commands the robot to sense distance. The robot responds, for

example, with "50 cm." Then the programmer asks the robot move forward one step and asks for a sensor reading again. The robot responds "30 cm." This game repeats until the robot comes within 10 cm of the object and the programmer stops issuing commands since the robot has reached its target.

6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program:
        i. Display distance from an object in the correct format
        ii. Locate an object not directly in front of the robot
        iii. Follow the object once it has been found
        iv. Keep the distance between the object and robot at exactly 10 cm, moving closer or further as the object moves about
        v. Adjust the power so that the robot moves faster the further it is from the object
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Line Follower

1) **Purpose:** Make a robot that follows a black line on top of a white surface.
2) **Material and Variables:** Light sensor, intensity (input), direction (output)

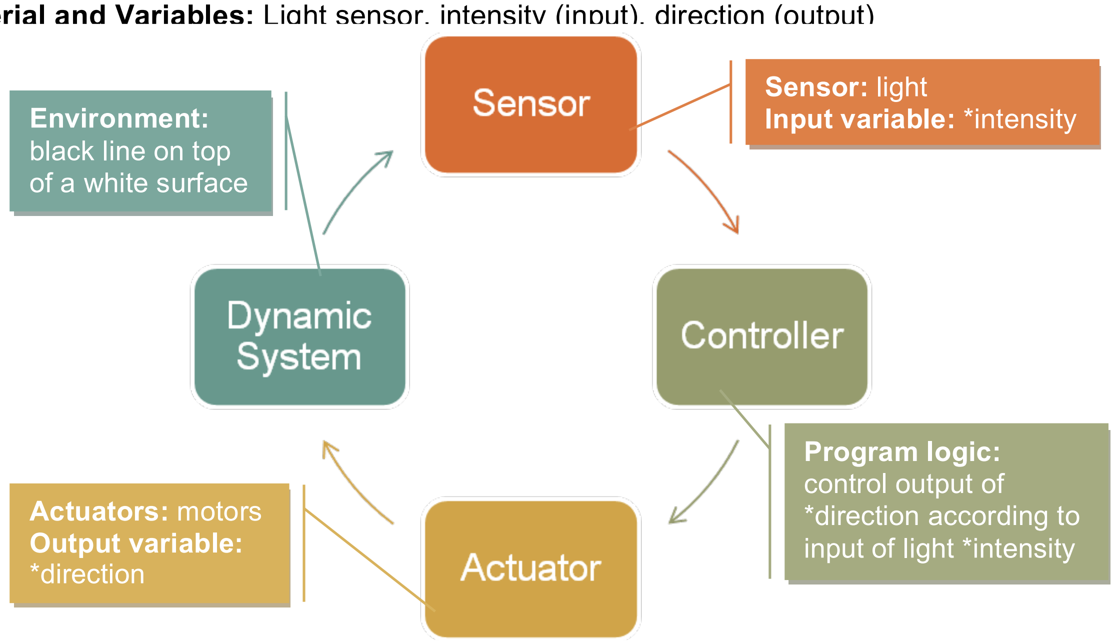


FIGURE 11: ROBOT LAYOUT DIAGRAM FOR LINE FOLLOWER

3) **Process Logic:**

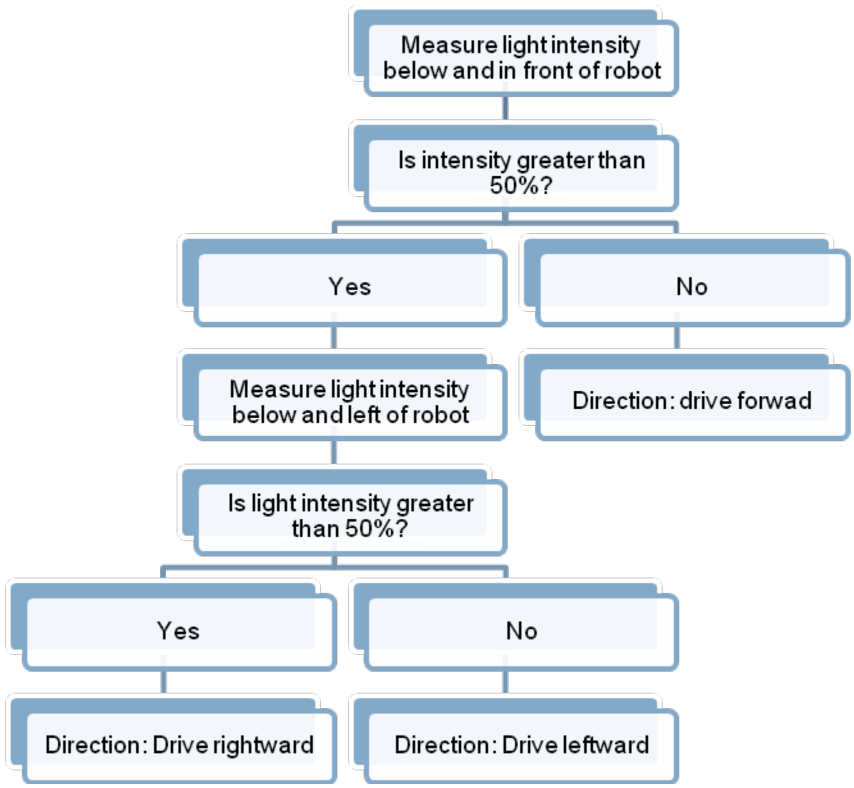


FIGURE 12: LOGIC FLOW CHART FOR LINE FOLOWER

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot while another is the programmer. The robot begins on top of a line drawn on the ground and waits for commands. The programmer, who cannot see what the robot is doing, commands the robot to sense the light intensity below it. The robot responds, for example, with "10%." Then the programmer tells the robot to move forward one step and asks for a sensor reading again. The robot responds "80%", to which the programmer commands the robot to turn one step to the left and sense again. This game repeats until the robot navigates the entire length of the line or completes a closed loop.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program :
        i. Drive forward when robot detects a low light intensity and turn when it detects a high light intensity
        ii. Efficiently follow a looped line by only turning left (traveling counter-clockwise)
        iii. Efficiently follow a looped line no matter which direction the robot travels
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Photovore

This lab must use the Compare Block.

1) **Purpose:** Make a robot that moves toward the brightest light source in an empty room while playing a tone according to the intensity of the light.
2) **Material and Variables:** Light sensor, light intensity (input), direction (output), power (output), frequency (output)

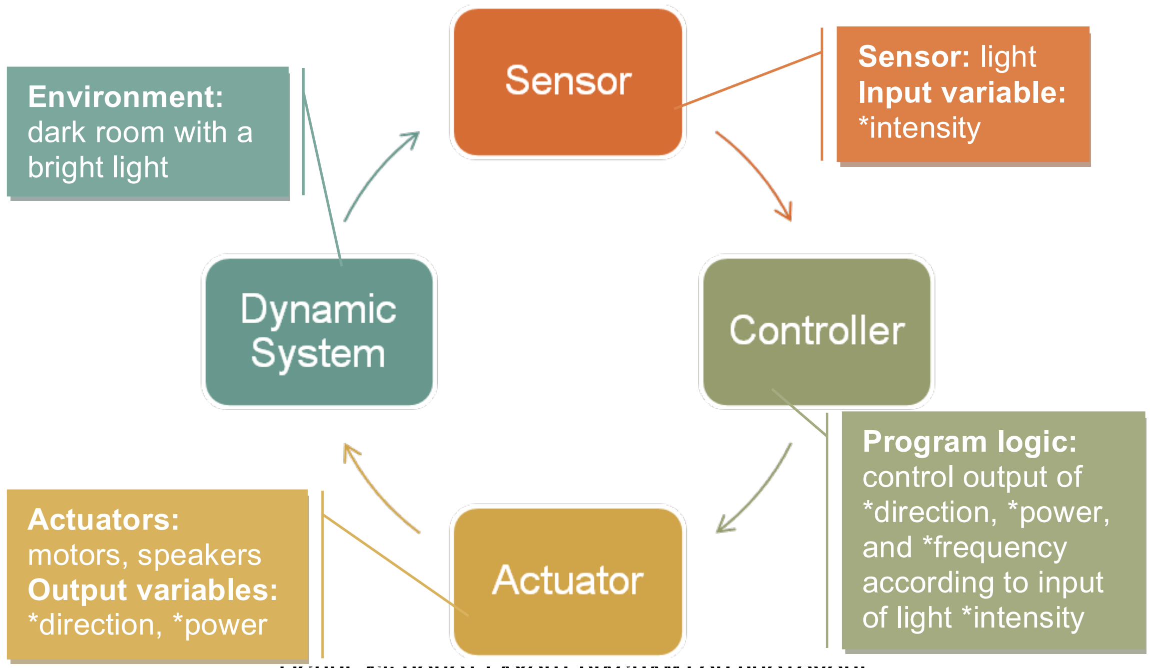


FIGURE 13: ROBOT LAYOUT DIAGRAM FOR PHOTOVORE

3) **Process Logic:**

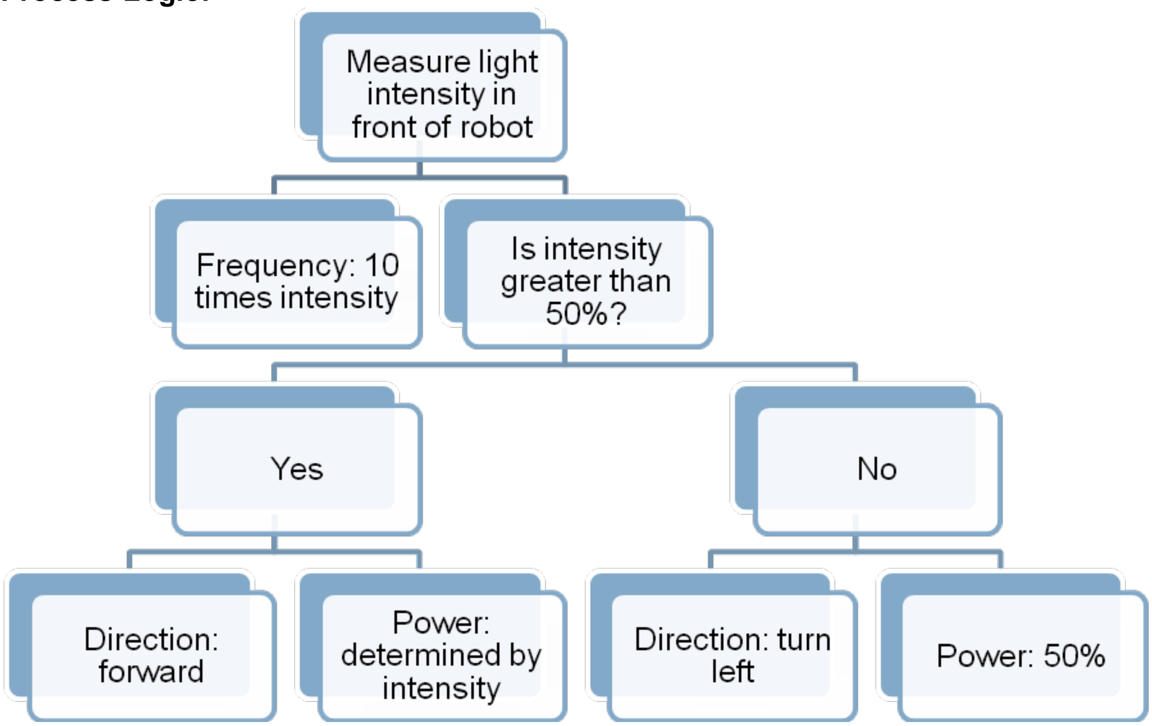


FIGURE 14: LOGIC FLOW CHART FOR PHOTOVORE

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot who awaits commands from a second student who is the blind programmer. A third student acts as a light source and is free to move about the room however he wants. The programmer commands the robot to sense the light intensity in front of it. The robot responds with "0%" when the light source is not in view. The programmer then tells the robot to turn one step to the left and sense the light intensity once more. This repeats until the robot is facing the light source and gives a sensor reading of "100%." The programmer then tells the robot to move forward one step before sensing again. This game repeats until the robot reaches the light source.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program:
        i. Play a tone at a frequency of 10 times the measured light intensity
        ii. Detect the light source by turning until it is directly in front of he robot
        iii. Drive toward the light source once the robot has detected it
        iv. Drive faster the further away the robot is from the light source
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Phototrope

This lab must use the Compare Block.

1) **Purpose:** Make a robot that moves away from the brightest light source in an empty room while playing a tone according to the intensity of the light.
2) **Material and Variables:** Light sensor, light intensity (input), direction (output), power (output), frequency (output)

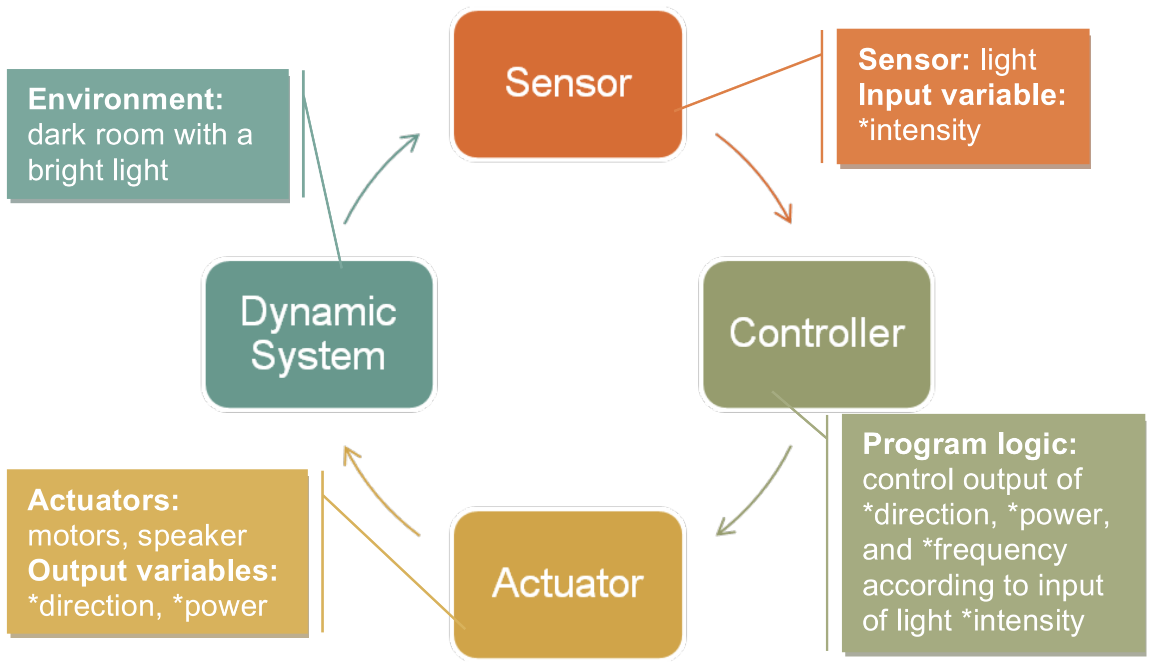


FIGURE 15: ROBOT LAYOUT DIAGRAM FOR PHOTOTROPE

3) **Process Logic:**

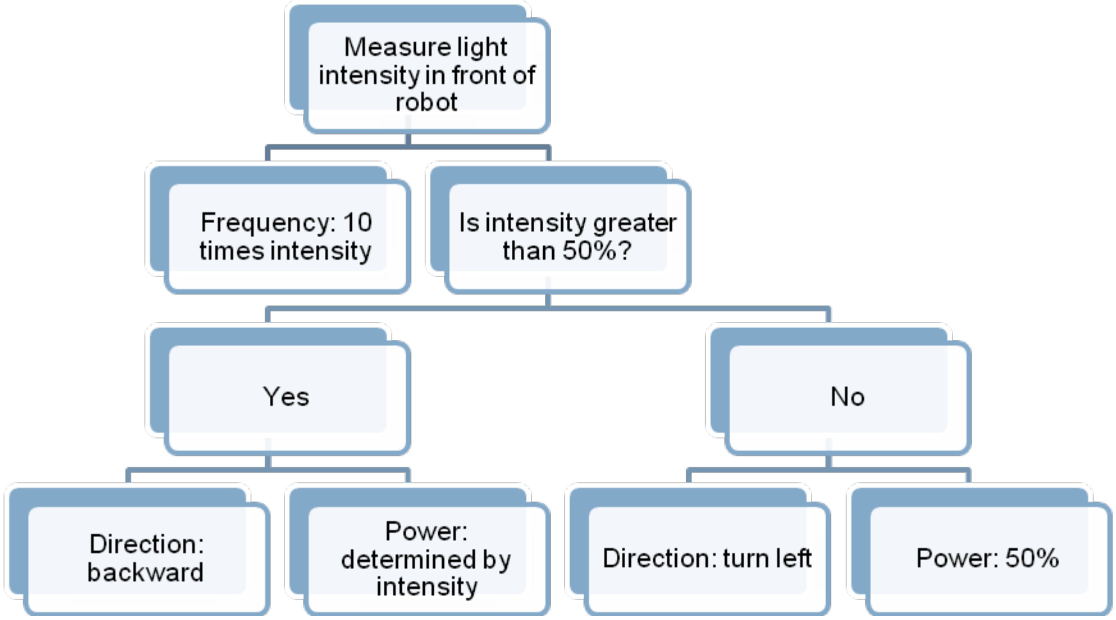


FIGURE 16: LOGIC FLOW CHART FOR PHOTOTROPE

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot who awaits commands from a second student who is the blind programmer. A third student acts as a light source and is free to move about the room however he wants. The programmer commands the robot to sense the light intensity in front of it. The robot responds with "0%" when the light source is not in view. The programmer then tells the robot to turn one step to the left and sense the light intensity once more. This repeats until the robot is facing the light source and gives a sensor reading of "100%." The programmer then tells the robot to move forward one step before sensing again. This game repeats until the robot reaches the light source.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program:
        i. Modify logic of Photovore program.
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Stopwatch

This lab cannot use Switch Blocks.

1) **Purpose:** Make a robot that acts as a stopwatch displaying the number of elapsed seconds once the NXT Enter button is pushed until it is pressed a second time to stop.
2) **Material and Variables:** NXT Enter button, timer, button Yes/No (input), time (input), display (output)

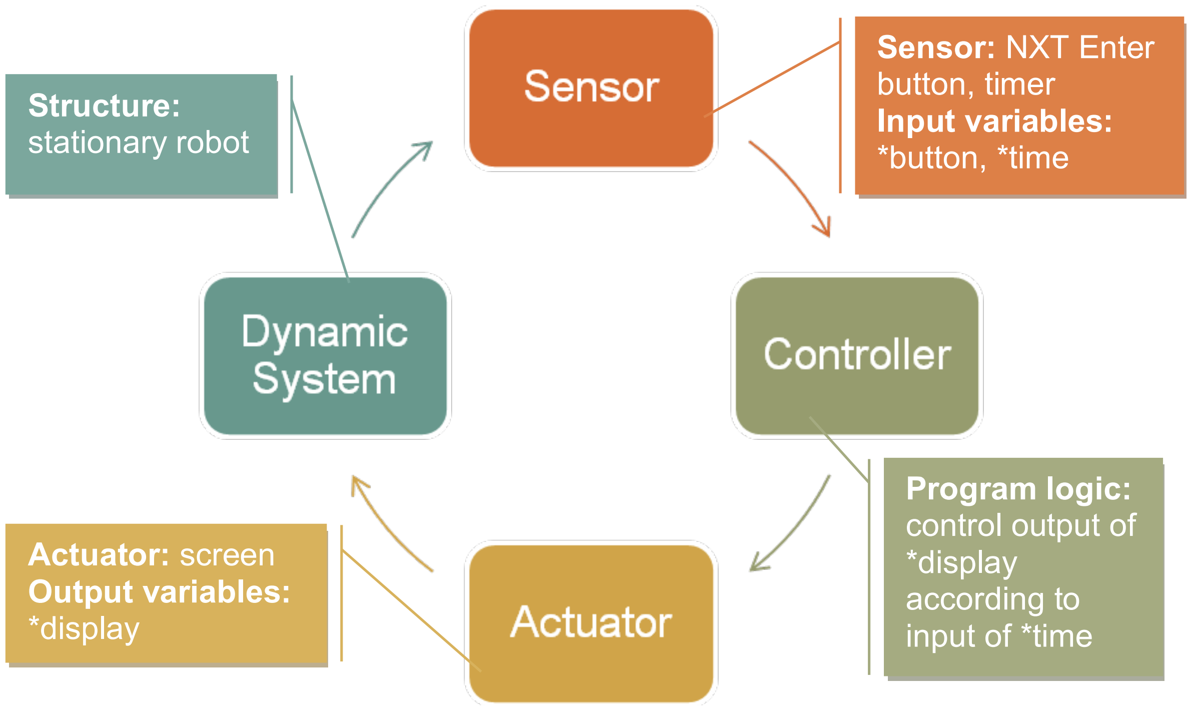


FIGURE 17: ROBOT LAYOUT DIAGRAM FOR STOPWATCH

3) **Process Logic:**

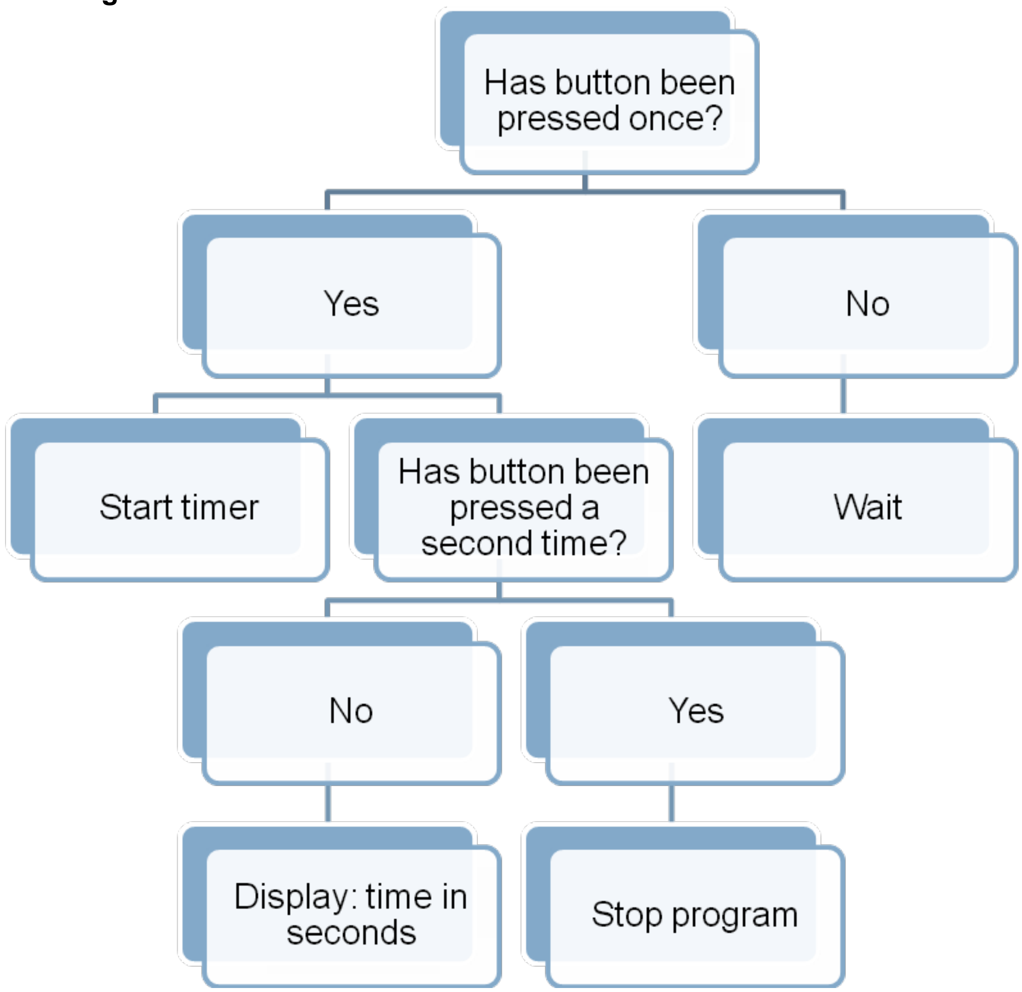


FIGURE 18: LOGIC FLOW CHART FOR STOPWATCH

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robotic stopwatch while a second student is the programmer. The programmer gives the robot a command for the NXT Enter button to be pressed. The stopwatch then starts counting in his head from 0. At random intervals the programmer asks the stopwatch for his current count, which the robot responds with. This continues until the programmer issues another command for the NXT Enter button being pressed.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program :
        i. Wait for the NXT Enter button to be pressed and then display the timer data
        ii. Convert timer data to seconds and stop the program when the NXT Enter button is pressed again
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Clapper

1) **Purpose:** Make a robot that moves based on the number of claps it counts before 5 seconds of silence.
2) **Material and Variables:** Sound sensor, timer, time (input), loudness (input), duration (output)

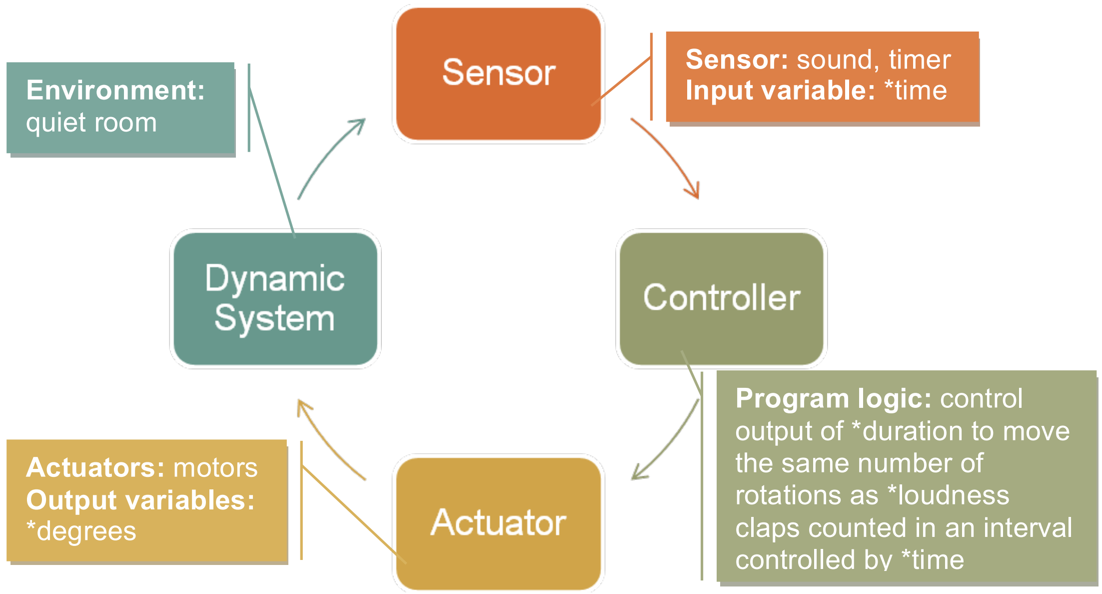


FIGURE 19: ROBOT LAYOUT DIAGRAM FOR CLAPPER

3) **Process Logic:**

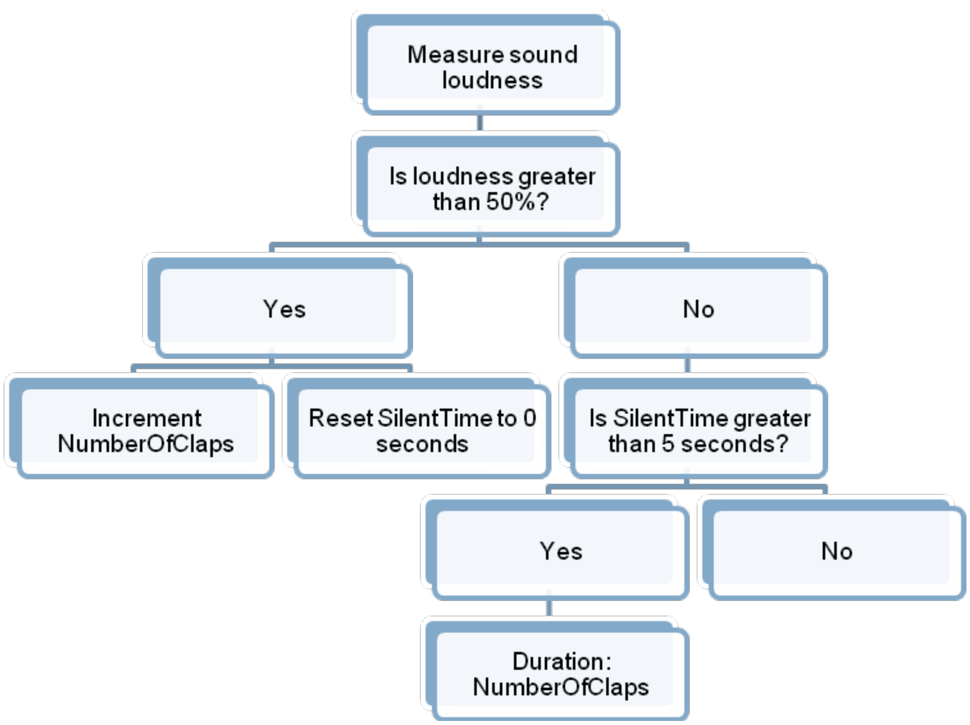


FIGURE 20: LOGIC FLOW CHART FOR CLAPPER

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot while another is the programmer. A third student performs the clapping. The programmer repeatedly asks the robot if it hears a clap. When the robot responds "Yes," the programmer instructs the robot to increment the total number of claps it has heard. When the robot responds "No," the programmer asks the robot if it has been 5 seconds since the last clap that was heard. The robot answers "No," and the game continues until it has been 5 seconds since the last clap. At this point the programmer asks the robot for the total number of claps that were heard and commands the robot to move forward that many steps.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program:
        i. Signal each time a clap is heard (whether by beeping every time it records a clap or displaying the current count on the screen)
        ii. Stop counting claps after 5 seconds of silence and display the total number of claps counted
        iii. Move a number of rotations equal to the number of claps counted before the 5 seconds of silence
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Additional activities

a) To reinforce that sensors are good for getting outputs and actuators are useful for taking inputs, the student can beep every time after a clap is recorded at the loudness measured by the sound sensor. In this exercise, the sound sensor provides the loudness measurement, and the actuator is the speaker that plays the beeping sound.

Random Numbers

1) **Purpose:** Make a robot that plays different sounds according to a randomly generated number.
2) **Material and Variables:** number (input), sound (output), display (output)

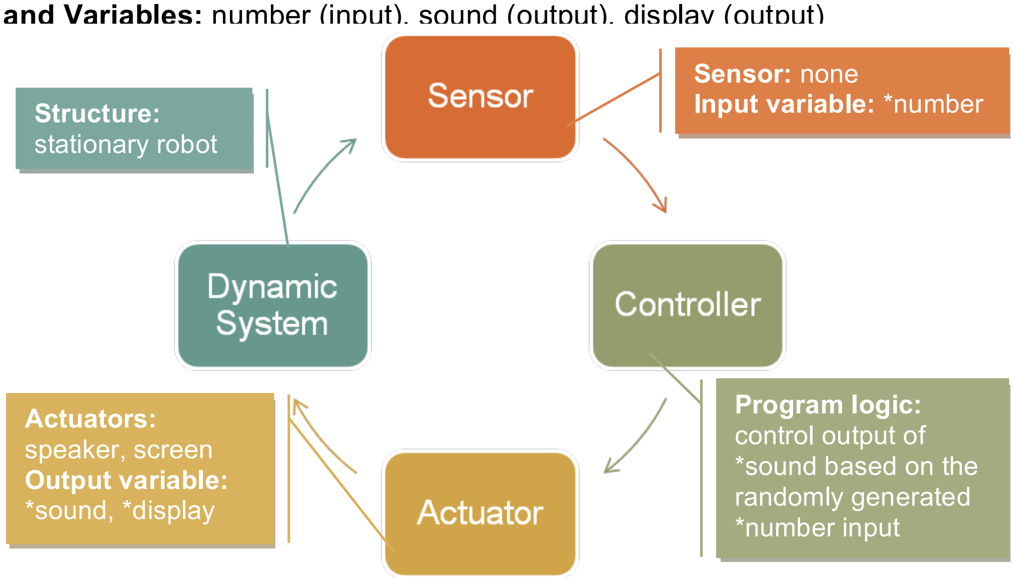


FIGURE 21: ROBOT LAYOUT DIAGRAM FOR RANDOM NUMBERS

3) **Process Logic:**

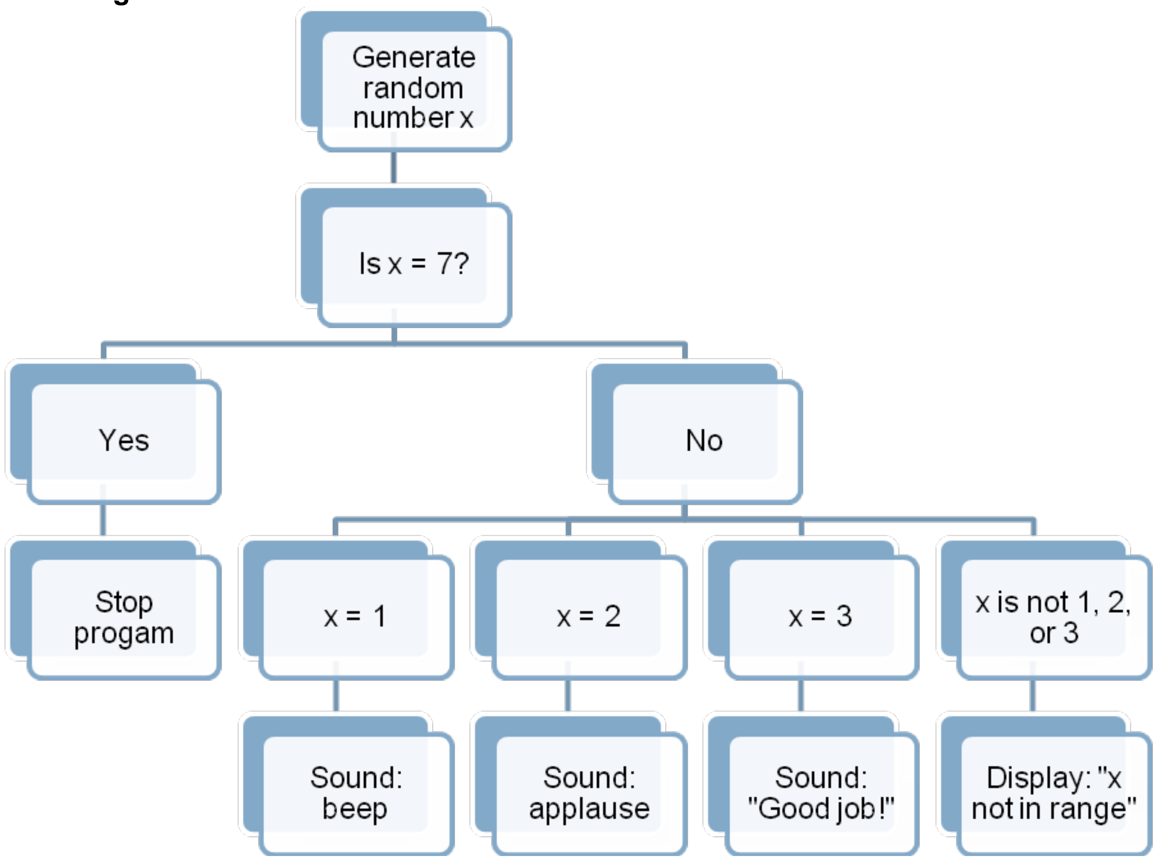


FIGURE 22: LOGIC FLOW CHART FOR RANDOM NUMBERS

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot while another is the programmer. A third student comes up with random numbers in the appropriate range. The programmer requests a random number and based on the specific number commands the robot to make a different sound. This continues until the programmer requests a random number and is given 7.
6) **Checkpoints:**
   a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
   b. Human Robot Game
   c. Build
   d. Program:
      i. Wait for NXT Enter button to be pressed then display a random number from 1–7
      ii. Loop (i) until the randomly generated number is not in the range 1–6 (students must use a Range block)
      iii. Switch on the number and play a beep if it is a 1, applause if it is a 2, and “Good job!” if it is a 3
      iv. Add a default case that displays “x is not in range” where x is the randomly generated number
   e. Test and debug
   f. Present and demonstrate robot to class or teacher

Combine

1) **Purpose:** Make a robot that executes one of the previous exercises according to the number of times the touch sensor is pressed:
   a. 1 press, execute "Stopwatch" exercise
   b. 2 presses, execute "Random Numbers" exercise
   c. 3 presses, execute "Clapper" exercise
2) **Material and Variables:** Touch sensor, timer, time (input), pressed Yes/No (input), display (output), all materials and variables contained in the "Clapper," "Stopwatch," and "Random Number" exercises

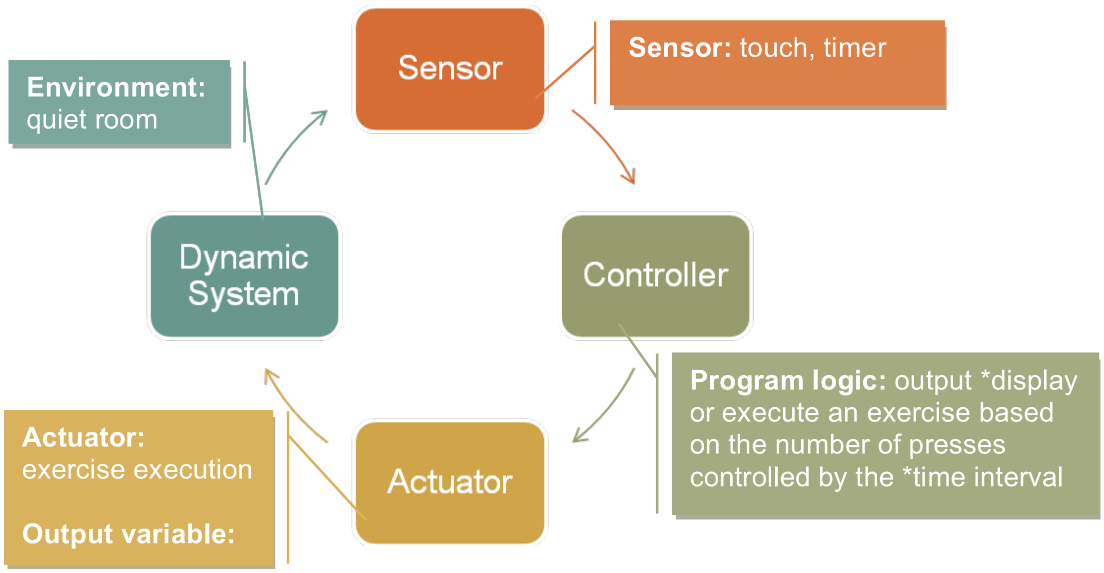


FIGURE 23: ROBOT LAYOUT DIAGRAM FOR COMBINE

3) **Process Logic:**

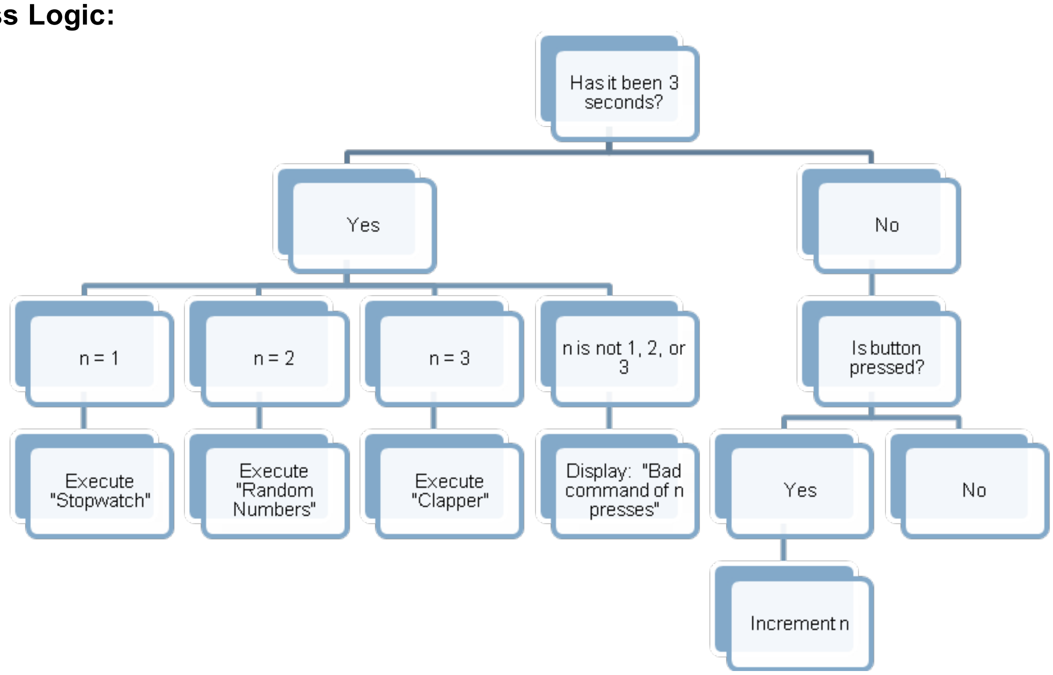


FIGURE 24: LOGIC FLOW CHART FOR COMBINE

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot while a second acts as the programmer. A third student is used to give touch sensor presses to the robot. The programmer asks the robot if it has been 3 seconds yet, to which the robot responds "No." The programmer then asks if the touch sensor is being pressed by the third student. When the robot responds "Yes," the programmer commands the robot to increment its counter. This continues until the robot responds "Yes" when asked if it has been 3 seconds, at which point the programmer asks the robot for the value of the counter. Based on the value, the programmer tells the robot which exercise to execute or message to display.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program:
        i. Display number of touch sensor presses that occur in first 3 seconds
        ii. Execute exercise program according to the number of presses counted
        iii. Add default case for when the number of presses is not 1, 2, or 3
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Differential Drive

This lab cannot use Move Blocks.

1) **Purpose:** Make a robot that maneuvers a known course by controlling each wheel's speed individually and using their circumferences to calculate distances.
2) **Material and Variables:** Rotation sensor, left-degrees (input), right-degrees (input), left-power (output), right-power (output)

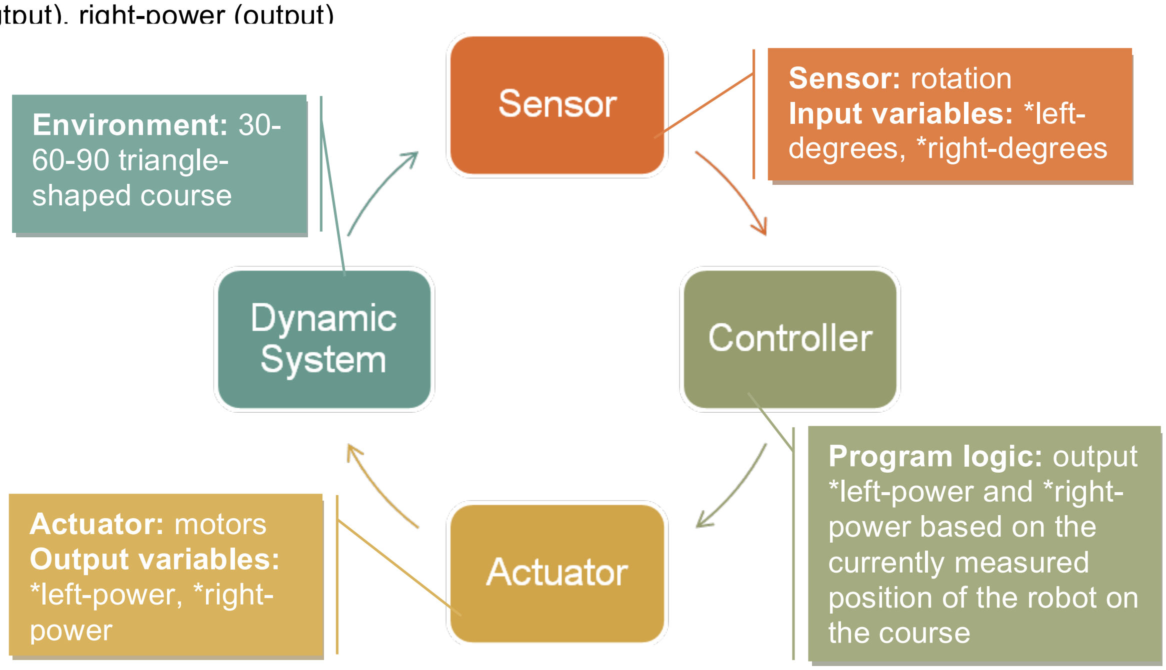
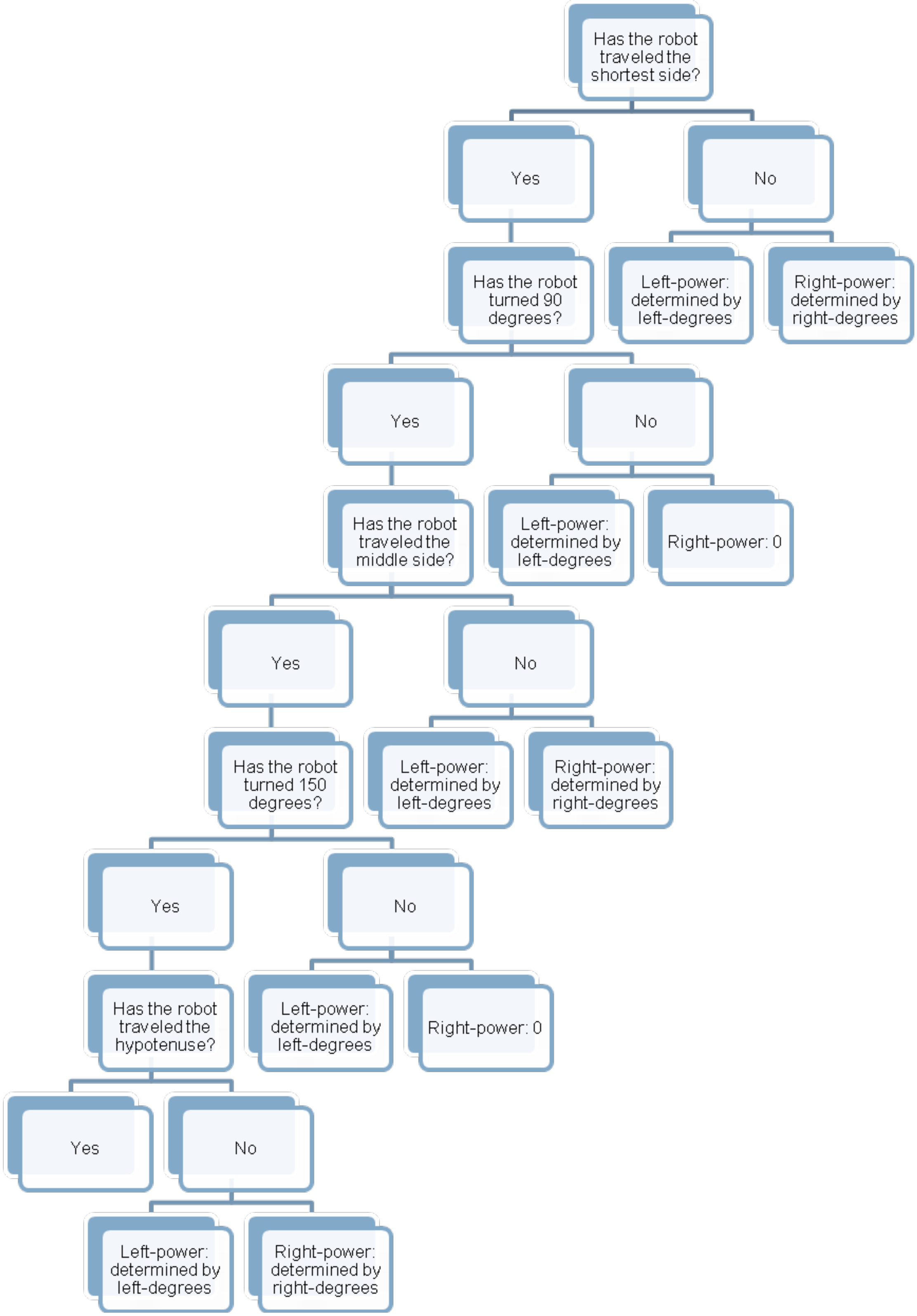


FIGURE 25: ROBOT LAYOUT DIAGRAM FOR DIFFERENTIAL DRIVE

3) **Process**



Logic:

FIGURE 26: LOGIC FLOW CHART FOR DIFFERENTIAL DRIVE

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot while a second acts as the programmer. The programmer asks the robot how many steps each of its feet have taken. The robot responds with the number, and if it they are fewer than the length of the current side the programmer tells the robot to move both feet one step forward. Otherwise, if the robot has taken enough steps to travel the side, the programmer commands the robot to keep its left foot stationary and pivot about it by moving its right foot forward. The programmer keeps doing this until the number of steps taken by the robot is enough to complete the first turn (as determined by how far apart the robots' feet are). The game continues for each side of the triangular course until the robot returns to its starting location.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart
    b. Human Robot Game
    c. Build
    d. Program:
        i. Measure the length of the shortest side and have the drive that distance by using Motor blocks and Rotation Sensor blocks in conjunction with the wheel diameter.
        ii. Modify the motor speeds in (i) to drive faster the more rotations it has left to do.
        iii. Measure how far apart the robots' wheels are and determine the arc length of a quarter-circle with that radius, then have the robot turn that amount after driving the length of the shortest side.
        iv. Measure the length of the middle side and drive that distance, going faster the more rotations the robot has left.
        v. Determine the arc length covering 150 degrees of a circle and have the robot turn that amount after driving the length of the middle side.
        vi. Measure the length of the hypotenuse and drive that distance to return to the start, going faster the more rotations the robot has left.
    e. Test and debug
    f. Present and demonstrate robot to class or teacher

Obstacle Avoidance

1) **Purpose:** Make a robot that avoids obstacles by moving away from those in front of it before contact and moving away from those behind it after contact.
2) **Material and Variables:** Ultrasonic sensor, touch sensor, distance (input), bumped Yes/No (input), direction (output) steering (output)

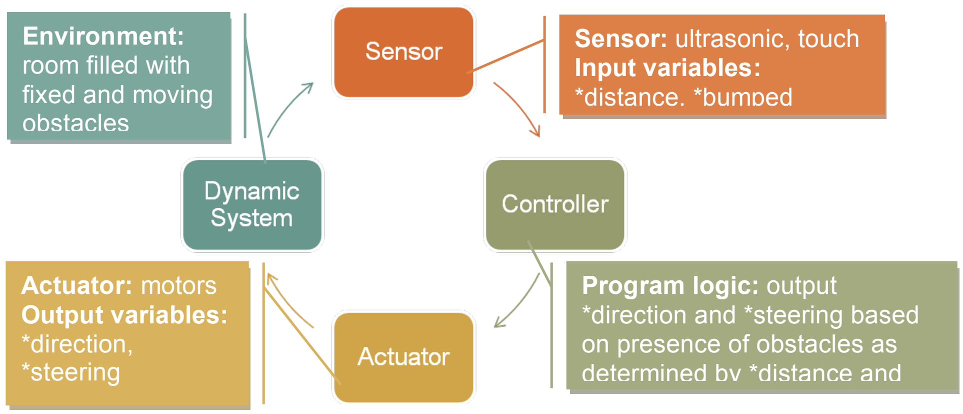


FIGURE 27: ROBOT LAYOUT DIAGRAM FOR OBSTACLE AVOIDANCE

3) **Process Logic:**

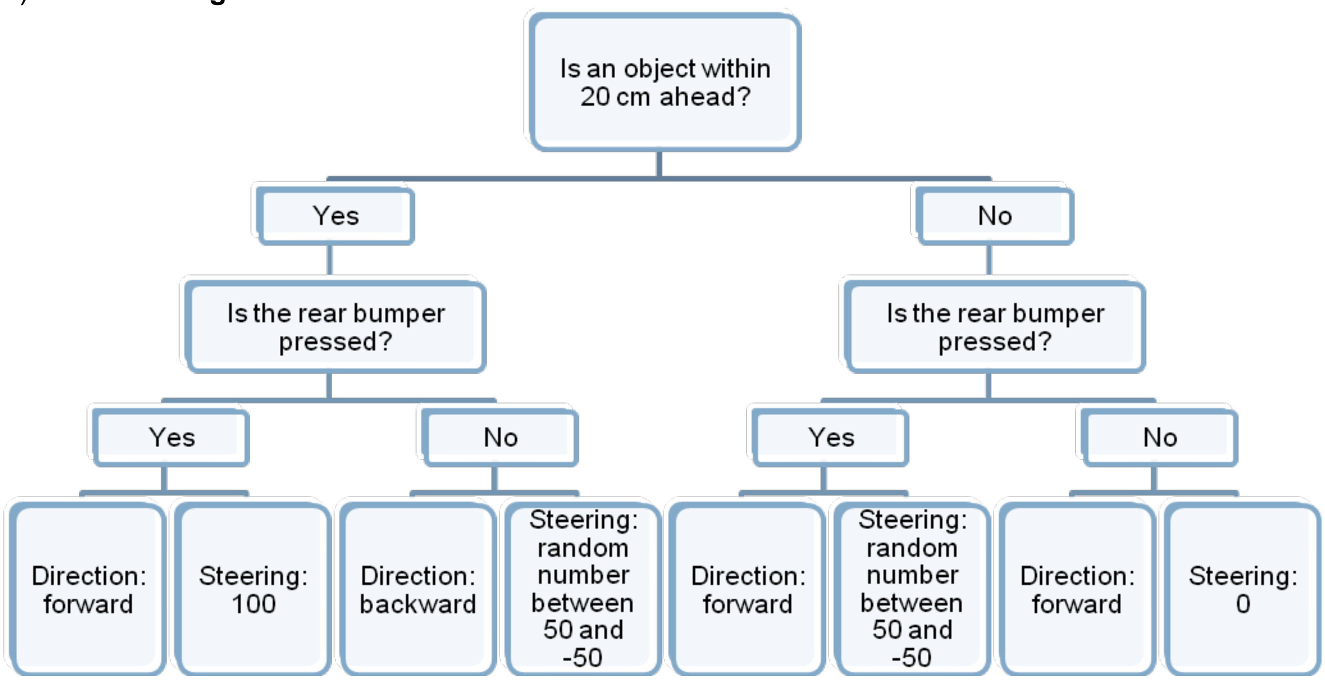


FIGURE 28: LOGIC FLOW CHART FOR OBSTACLE AVOIDANCE

4) **Assign Roles**
5) **Human Robot Game:** One student acts as the robot while a second acts as the programmer. The other students on the team are the obstacles, some of which can move around freely. The programmer asks the robot if there is an obstacle in front of it. The robot responds “No,” and the programmer then asks if it is touching an obstacle behind it. The programmer commands the robot to take one step forward and to the side when the robot responds “Yes” and commands the robot to take one step directly forward when the robot responds “No.” This continues with the programmer again asking if there is an obstacle in front of the robot. The robot responds “Yes,” and the programmer then asks if it is touching an obstacle. The programmer commands the robot to spin in place when the robot responds “Yes” and to take a step backward and to the side when the robot responds “No.” This game continues for a pre-determined amount of time.
6) **Checkpoints:**
    a. Purpose definition, Robot Layout Diagram, and Logic Flow Chart

b. Human Robot Game
c. Build
d. Program:
    i. Drive forward when there is no obstacle in front of the robot and backwards at an angle when there is an obstacle in front of it.
    ii. Modify (i) to also spin in place when there is an obstacle in front of the robot and the bumper is being pressed.
    iii. Modify (ii) to drive forward at an angle when there is no obstacle in front of the robot and the bumper is being pressed.
e. Test and debug
f. Present and demonstrate robot to class or teacher

Text Messenger

This lab exercise is taken from one of our resources.[7] As we could not test it completely during the workshop since it requires more than one NXT for communication, a complete walkthrough guide for this lab is included.

Purpose: Establish a Bluetooth connection between two NXTs and use one to send a text message to the other.

Materials: 2 NXT Intelligent Bricks

Activate Bluetooth:

1. On the NXT's menu, scroll over to the Bluetooth option using the arrow buttons.
2. Select Bluetooth by pushing the orange button.
3. Scroll over to On/Off and select that menu option.
4. Turn the Bluetooth On. When you select On, a little Bluetooth symbol will appear in the upper left-hand corner of the screen.

Rename an NXT:

1. In NXT-G, select your NXT window icon in the bottom-right corner control panel.
2. Select your NXT under your communication tab.
3. In the right-hand corner is a box to rename your NXT. Type in a unique name and then click the arrow to the bottom right when done.

Establish Bluetooth Connection between NXTs:

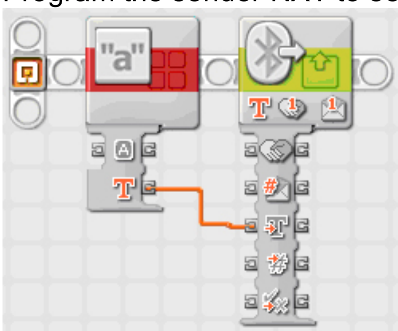
1. Return to the Bluetooth option on the NXT menu.
2. On the sender NXT, scroll over to Search.
3. The NXT will then prompt you to choose a slot (the NXT can be connected to up to 4 devices at a time). Choose a free slot.
4. Each NXT will beep and you will be prompted to enter a passkey. The passkey is “1234” and should be automatically entered on the screen.

[7] http://legoengineering.com/index.php?option=com_wrapper&view=wrapper&Itemid=83

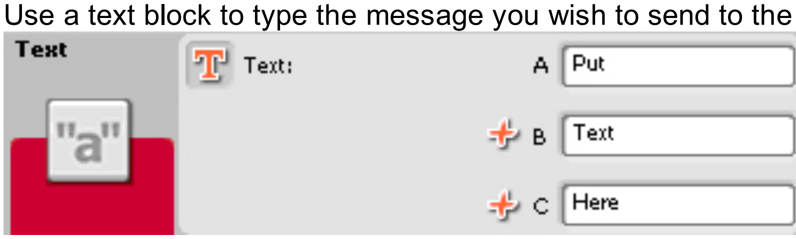
5. Select the checkmark on both NXTs to establish the connection. Connections can be verified by looking at the top-left corner of the NXT screen. If the Bluetooth symbol only has a “<” sign next to it, no connection has been made. If it has “<>”, then a connection is made.
6. Once connected, you can view your available connections under the Bluetooth menu. Scroll over to Connections to view.

Programming Instructions:

1. Turn the Bluetooth feature on for both NXTs.
2. Rename each NXT so that they can easily identify one another.
3. Choose one NXT to be the sender and one to receive the text message. Establish a connection between the NXTs using the sender.
4. Program the sender NXT to send a message via Bluetooth.

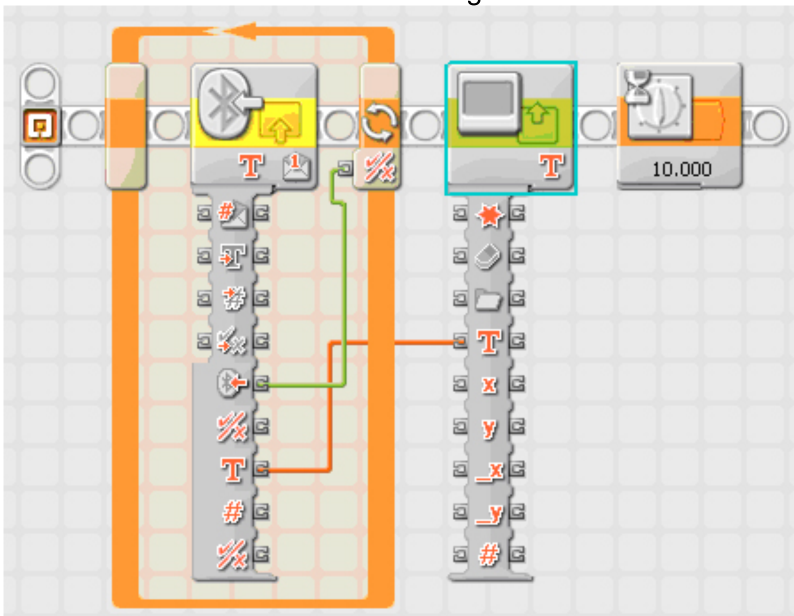


   a. Use a text block to type the message you wish to send to the receiver NXT.

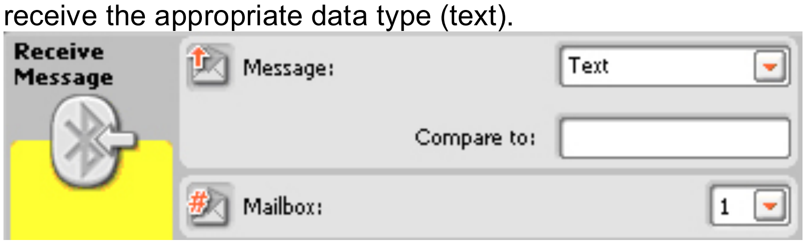


   b. Use a Send Message block to select the connection number (previously designated when establishing the connection) and data type (text in this instance).

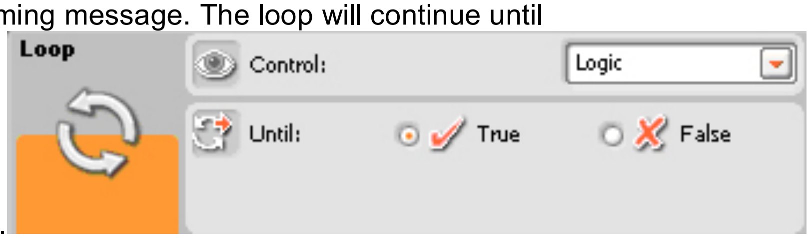
5. Program the receiver NXT to receive a message via Bluetooth.



   a. Use a Receive Message block to monitor for an incoming message. Set the block to receive the appropriate data type (text).

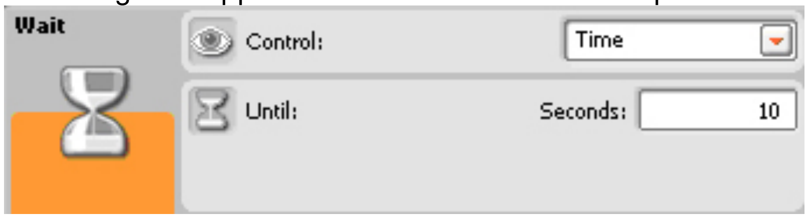


   b. Place the Receive Message block in a loop so that it can continuously monitor for the incoming message. The loop will continue until True.



   c. Use a Display block so that the received message will appear on the NXT screen.

d. Use a Wait block so that the message will appear on the screen for a certain period of time.



APPENDIX E: NXT-G PROGRAMMING BLOCKS

Common

- Move
  - Ports
  - Direction
  - Steering
  - Power
  - Duration
  - Next Action
  - Ways to use connections
- Record/Play
  - Action
  - Name
  - Recording
  - Time
  - Ways to use connections

Action

- Motor
  - Port
  - Direction
  - Steering
  - Power
  - Duration
  - Wait for completion
  - Next Action
- Sound
  - Sound File
  - Tone
  - Volume
  - File

  - Wait for completion
- Display
  - Action
  - Clear
  - File
- Send Message
  - Connection
  - Message
  - Mailbox

Sensors

- Touch Sensor
  - Action
- Ultrasonic Sensor
  - Distance
  - Trigger Point
  - Greater/Less
  - Yes/No
- Light Sensor
  - Generate Light
  - Light Intensity
  - Trigger Point
  - Greater/Less
  - Yes/No
- Sound Sensor
  - Sound Level
  - Trigger Point
  - Greater/Less
  - Yes/No
- NXT Buttons
  - Button
  - Action
- Rotation Sensor
  - Action

- Trigger Point
- Greater/Less
- Yes/No
- Degrees
- Direction

- Timer
  - Timer number
  - Action
  - Yes/No
  - Value
- Receive Message
  - Message
  - Mailbox
  - Message Received

Flow

- Wait
  - Time
  - Sensor
- Loop
  - Count
  - Sensor
  - Time
  - Forever
  - Logic
- Switch
  - Sensor
  - Value
  - Number, text, logic
- Stop

Data

- Logic
  - And

  - Or
  - Xor
  - Not
- Math
  - Addition
  - Subtraction
  - Multiplication
  - Division
- Compare
  - Greater than
  - Less than
  - Equals
- Range
  - Yes/No
- Random
  - Range
- Variable
  - Type
  - Action
  - Value
  - Creating new variables

Advanced

- Text
  - Concatenation
- Number to Text
- Calibrate
  - Port
  - Sensor
  - Action
  - Value

APPENDIX F: THE HUMAN ROBOT

This activity is a simple simulation of a robot that provides students with an understanding of what is involved in designing software to control robots. The general model usually involves a brain or a programmer that directs the entire system with a restricted set of commands and a robot that can only understand these words. The robot will respond to the programmer with information accordingly so that the programmer can proceed with the next action.

There are two models to play this game. The first one has one student representing the complete robot and the second one uses a plural number of students to represent the robot. In both of these models there is only a single brain or programmer.

Model 1: One Human as One Robot

One student acts as the robot. It is essential that this person is “blind” in the sense that he does not respond to the environment without being issued commands. Closing the robot’s eyes or using a blindfold can help reinforce this concept. Another student acts as the programmer and must have his back facing the robot at all times. The programmer will command the robot with a restricted set of commands. The robot can only respond to those. The robot will give data and information by saying it aloud. The programmer will listen for feedback from the robot and keep commanding the robot until the task is accomplished.[8]

Model 2: Many Humans as a Robotic System

This second model is closer to how robots are actually programmed. The programmer usually worries about the behavior of the robot by considering each subsystem as a separate entity. These are commonly subdivided into the sensors, actuators, and dynamic system. This model also has a single programmer acting as the controller, but the parts of the robots are assigned to many different students. For example, one student could be the eyes, another one the legs, and a third the bumpers.

An example of this is explained in the following activity that simulates a robot using seven people linked together in a group, each mimicking a different component of the robot's behavior by following a precise set of instructions. When the independent instructions of each component are followed properly and precisely, the robot (a composite of the seven people) performs the desired actions. In this way, the participants can comprehend the resulting behavior of the whole robot as an integrated set of interdependent smaller actions.[9]

The following is a schematic of the robot's configuration:

[8] http://www.education.rec.ri.cmu.edu/roboticscurriculum/curriculum/introductiontosoftwareandelectroniccontrol.htm

[9] http://courses.washington.edu/art483/site/pages/livingrobotsim.shtml

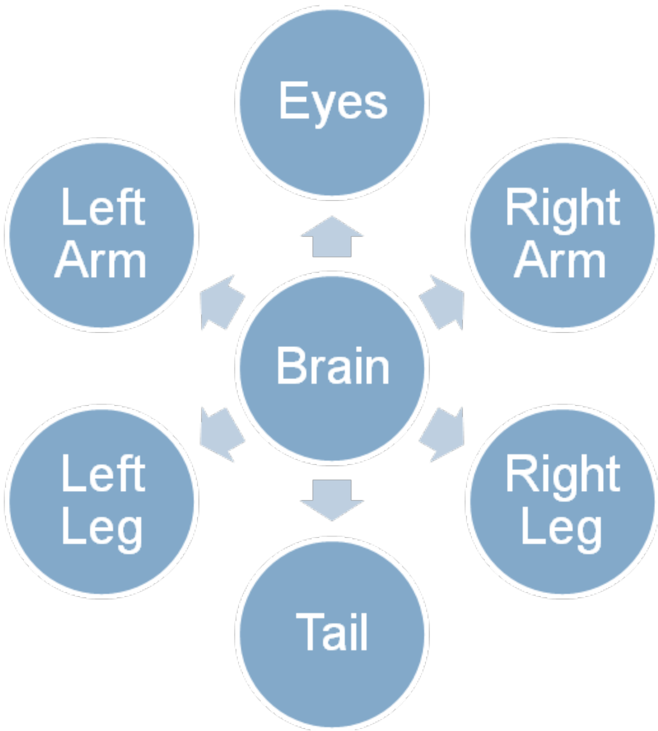


FIGURE 29: ROBOTIC SYSTEM WITH SEVEN STUDENTS

If seven students are too large of a group, this can be rearranged to work with smaller numbers. For example, a group of four students could be:

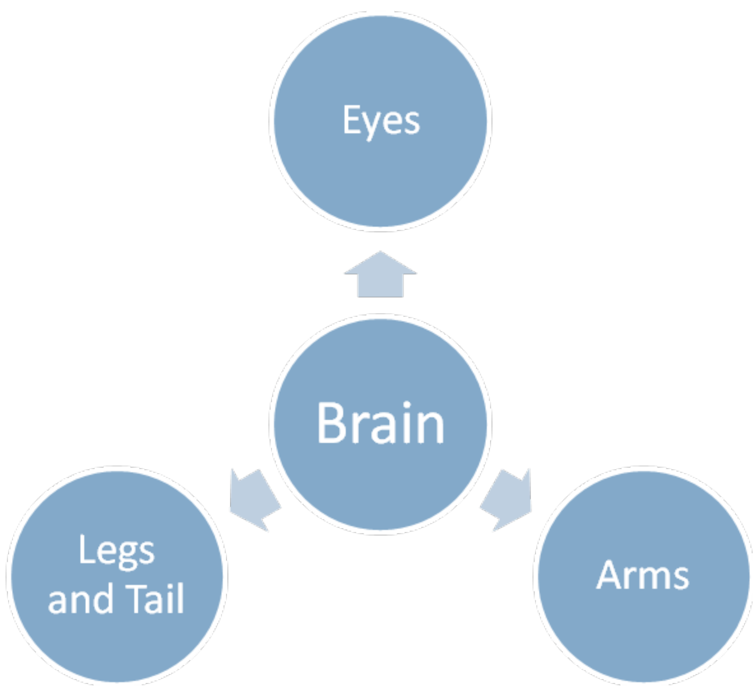


FIGURE 30: ROBOTIC SYSTEM WITH FOUR STUDENTS

The functions of the seven components are as follows:

- **Brain:** controls all operations and actions of the other components
- **Eyes:** senses the intensity of the ambient light in front
- **Left Arm:** senses physical contact with an object on the left side
- **Right Arm:** senses physical contact with an object on the right side
- **Left Leg:** propels the left side of the robot forward or backward
- **Right Leg:** propels the right side of the robot forward or backward
- **Tail:** signals with a rear-facing lamp

Using this configuration, the robot moves steadily forward until a light is shined on it, at which point it stops. While moving, if it is bumped on the right side, it will turn to the left and turn on the tail lamp during the turn. Similarly, if bumped on the left, it will turn to the right with the lamp likewise indicating this.

In order to accomplish this, the Brain must synchronize the activities of all the other components. It continuously monitors the information arriving from the input sensors—the Eyes, Left Arm, and Right Arm. Processing that information, it then directs the activities of the actuators—the Left Leg, Right Leg, and Tail.

The following table presents the instructions for each component of the Human Robot.

TABLE 8: HUMAN ROBOT COMPONENT INSTRUCTIONS

Component	When you…	Do this…
Left Arm	contact anything	say *LEFT CONTACT*
Right Arm	contact anything	say *RIGHT CONTACT*
Eyes	see a bright light	say *BRIGHT LIGHT*
Left Leg	hear *LEFT LEG GO* hear *LEFT LEG STOP*	walk slowly forward stop walking
Right Leg	hear *RIGHT LEG GO* hear *RIGHT LEG STOP*	walk slowly forward stop walking
Tail	hear *LAMP ON* hear *LAMP OFF*	turn on light turn off light

The brain is a little more complicated because it has to orchestrate many actions based on input from the other components:

TABLE 9: HUMAN ROBOT BRAIN INSTRUCTIONS

Component	**When you…**	**Do this…**
Brain	want to start the robot	say *LAMP OFF* say *LEFT LEG GO* say *RIGHT LEG GO*
	hear *BRIGHT LIGHT*	say *LAMP OFF* say *LEFT LEG STOP* say *RIGHT LEG STOP*
	hear *LEFT CONTACT*	say *LAMP ON* say *RIGHT LEG STOP* wait one second say *RIGHT LEG GO* say *LAMP OFF*
	hear *RIGHT CONTACT*	say *LAMP ON* say *LEFT LEG STOP* wait one second say *LEFT LEG GO* say *LAMP OFF*

APPENDIX G: ADDITIONAL RESOURCES

Textbooks

The following list presents some textbooks that can be used to learn foundational material in subjects related to robotics. It is recommended that the faculty involved in teaching robotics have a mastery of all of these.

Electronics

The Art of Electronics – One of the most complete and thorough textbooks that contains practical and comprehensive material on electronic circuit behavior and design.

Mathematics

Calculus Early Transcendental – Includes material covering single variable calculus (differentiation and integration) as well as multivariable calculus.

Elementary Differential Equations and Boundary Value Problems – Covers topics in differential equation classes.

Mathematical Statistics with Applications – Covers probability and statistics as applied in engineering and scientific settings.

Physics

University Physics – University level calculus-based physics (both mechanics and electromagnetism).

Online Resources

The following websites on robotics education were obtained from RoboticsCourseWare. These may be of interest to robotics faculty members.

External Robotics Course Materials

OpenCourseWare Finder – Provides search functionality to registered OpenCourseWare courses. Search "robotics" for links to a number of registered courses, mostly from MIT.

Robotics Curriculum Clearinghouse – A NASA project that archives links to robotics teaching materials for primary and secondary schools.

Institute for Personal Robots in Education – Organization dedicated to applying and evaluating robots within the context of computer science education.

Teaching Resources of the EU Cognitive Systems Outreach Program – Provides links to publicly-accessible teaching materials for robotics and cognition-related courses.

Stanford Engineering Everywhere – Links to course materials including lecture videos for some of Stanford University's engineering courses. Includes Introduction to Robotics by professor Oussama Khatib.

Video Lectures for Robotics at IIT Bombay – Indian Institute of Technology Bombay videos of robotics lectures.

Cognitive Robotics Course at CMU – Materials for Carnegie Mellon University's Cognitive Robotics course taught by professor Dave Touretsky. Uses Chiara robot hardware and Tekkotsu software packages linked below.

Free Robotics Texts

Planning Algorithms – Free textbook by Steven M. LaValle on many of the planning problems encountered in robotics.

Introduction to Computer Science via Robots – Free textbook by the Institute of Personal Robots in Education.

Robotics Sites and Message Boards

Lugnet – Message board for LEGO Mindstorms users and enthusiasts. Includes well-organized and archived LEGO robotics information.

Robot Cafe – Well-organized robotics directory that includes various software, hardware, and robotics competition links.

Robotics Hardware

E-Puck – Open-hardware education-oriented mobile robot developed by EPFL (Lausanne, Switzerland).

LEGO Mindstorms – LEGO sets with programmable bricks. Can be used with a variety of programming languages including C, C++, Java, Python, VisualBasic, and LabVIEW.

Handyboard – 6811-based microcontroller system that can be used for building mobile robots for educational, hobbyist, and industrial purposes.

Make Controller Kit – Open source hardware solution for hobbyists and professionals with numerous robotics applications. Supports desktop interfaces via a variety of languages including Max/MSP, Flash, Java, Python, Ruby, and anything that supports OSC.

Chiara Open Source Educational Robot – Open source educational robot hardware solution called Laptop with Legs. Also provides designs for iRobot Create platform and planar hand-eye system for use with Tekkotsu software platform listed below.

Robotics Software and Libraries

OpenRave – Open source cross-platform plugin-based robot planning architecture that serves as the center for all planning and execution. Includes services such as collision detection, robot kinematics, physics, robot controls, and a network scripting environment.

Pyro – Project designed to create an easy-to-use interface for accessing and controlling a wide variety of real and simulated robots.

RobotML – Experimental XML-based markup language used for communication between autonomous mobile robots and robot components.

Player Project – Project to create free software for research into robotics and sensor systems. Components include the Player network server and Stage and Gazebo robot platform simulators. These run on POSIX-compatible operating systems including Linux, Mac OS X, Solaris, and the BSD variants.

RubiOS – Java-based open source communication API for Social Robots.

ARIA – Open source robot API libraries compatible with Pioneers, Peoplebot, Seekur, Amigobot, Powerbot, and other robots. Programmable with Java, C++, and Python. Provides tools to integrate input/output capabilities with your own custom hardware.

CLARAty – Collaborative effort among NASA Jet Propulsion Laboratory, NASA Ames Research Center, Carnegie Mellon University, and the University of Minnesota to promote reusable robotics software. Designed to support heterogeneous robotics platforms and integrate advanced robotics capabilities from multiple institutions.

ERPS – Commercial robotics software development kit.

Experimental Robotics Framework – Add-on for Player and Stage that supports multiple robot experiments in three dimensions.

MARIE – Free software tool using a component-based approach to build robotics software systems by integrating previously-existing and new software components.

Microsoft Robotics Studio – Uses a component-based approach where each component is a service. Services are orchestrated through a concurrency library called the CCR. Current version includes a Microsoft XNA-Framework–based simulator with physics simulation and a visual programming language.

Mobile Robot Programming Toolkit – Open source set of C++ libraries and applications which cover grabbing, visualizing, and manipulating datasets, particle filter and Kalman filter-based simultaneous localization and mapping, linear algebra, robotics sensors, and MATLAB-like plot rendering.

OpenJaus – Open source component-based approach that uses standardized messages and transport methods to create interoperable robotic systems.

ORCA – Open source framework for developing component-based robotic systems. Provides the means for defining and developing the building blocks which can be pieced together to form arbitrarily complex robotics systems, from single vehicles to distributed sensor networks. Inter-component communication is implemented using Ice middleware.

Orchestra Control Engine – Suite of software components based on Linux/RTAI for the planning, development, and deployment of real-time control applications for industrial machines and robots.

Open Robot Control Software – Project that provides a free software toolkit for real-time robotic arm and machine tool control.

RoboMind – Educational software to learn the basics of robotics and programming.

Robot Intelligence Kernel – Portable reconfigurable suite of perceptual, behavioral, and cognitive capabilities that can be used across many different platforms, environments, and tasks. Integrates algorithms and hardware for perception, world-modeling, adaptive communication, dynamic tasking, and behaviors for navigation, search, and detection.

RoSta – European robotics software architecture and middleware approaches. Provides plenty of information on architectural patterns and styles, data logging and storage, and catalogs of robot software.

Skilligent – Commercial control system for autonomous service robots. Enables creation of robots that can be trained by end users rather than programmed by a software engineer.

URBI – Universal Real-time Behavior Interface.

Webots – Commercial robot simulation package that allows physically realistic simulations and allows prototyping your own robot. Comes with a broad palette of predesigned robot models (e.g., Aibo, E-Puck, Khepera, Hoap2, Hemisson, Pioneer).

Parallel Port PWM/Encoder Linux Driver – Cheap alternative for motor control. Allows open- and closed-loop control of motors using parallel ports.

LDraw – Three dimensional LEGO modeling software, as well as libraries of virtually every LEGO part ever made.

Tekkotsu – Open source robotics educational software from Carnegie Mellon University.