North Carolina School of Science and Mathematics

# EE4100 : Introductory Robotics

**Course Description:** This course provides students with the opportunity to develop skills in basic programming and design using an autonomous LEGO EV3 robot and the Python programming language. Students will explore the use of sensors to have the robot react to its environment and learn to troubleshoot mechanical and software issues. Self-guided skill development early in the semester is followed by a series of project challenges emphasizing teamwork and design.

**Instructor Name or Department:**
Larry Myers
Engineering and Computer Science

**Course Meeting Pattern:**
Residential - 4 days/week: 3 x 50 min + 1 lab x 90 min

**Prerequisites/Corequisites:**
None

**Textbook(s):**
None

**Required Course Materials (including technology):**

- This course is delivered via the Canvas Learning Management System.
- Students are expected to have a Windows or Mac computer to access instructions, write programs, and submit assignments
- LEGO MindStorms EV3 Core Set (45544) will be supplied. Students are responsible for returning an intact and inventoried LEGO kit.
- Python development environment in Visual Studio Code and Anaconda will be provided to be installed on student's computers.
- Robotics Project Challenge Mats will be supplied. Students are responsible for returning the mats in good condition.
- Tailored Python learning materials will be provided in Canvas (Based on PythonLikeYouMeanIt.com)
- Tailored EV3 MicroPython learning materials will be provided in Canvas (Based on LEGO Education EV3 MicroPython)

**Learning Outcomes:**
In this course, students will:

- Review the evolution of robotics and distinguish robots from simpler machines
- Learn the Python programming environment, including looping structures, decisions, comparison operators, and methods of code reuse
- Utilize a variety of sensors to incorporate environmental conditions into decisions governing a robot's behavior
- Design and build mechanical systems suited towards a specific goal
- Diagnose and rectify the sources of errors in programs and mechanical systems
- Use sensors to reduce error in desired outcomes and accommodate environmental variability
- Utilize the engineering design process to identify mechanical functions, sensors, and algorithms needed for a robot to achieve a specific goal
- Describe design intent, modifications, and system performance through project reports and presentations
- Work in teams and coordinate the design and implementation of a robots subsystems

Upon successful completion of this course, the student will be able to:

- Describe the differences between a robot and a simple machine
- Identify the various major disciplines involved in the study and development of robots
- Determine necessary sensors, mechanical systems, and algorithms to achieve a task
- Describe sources of sensor variability and define strategies to overcome imperfect data
- Identify if and what type of sensors are needed to achieve a function and the value of the sensor input over a strategy relying on dead reckoning
- Use sensory feedback in a control loop to maintain a desired goal under changing conditions
- Describe the difference between bang-bang and proportional control and how behavior changes with increasing proportional gain
- Write programs using the Python programming language
- Distinguish between types of error (offset, proportional, discretization, repeatability, precision) and how to identify these errors using observed data
- Choose gear ratios and wheel sizes to achieve high speed or high torque
- Create mechanisms to push, grasp, lift, and/or otherwise engage with objects
- Implement programing structures such as While loops, For loops, and If … Else statements
- Identify repetitive code and replace it with function calls and other means of code reuse
- Work successfully in teams to make design decisions, build a robot, and implement the algorithms in programming with a limited schedule and facilities
- Write reports documenting strategies, design ideas, implementation, and results incorporating text, annotated figures, and programming code
- Present design ideas and programming strategies succinctly to an audience

**Grading Policy:**
**Evaluation and Grading**

Your grade for the course will be calculated as follows:

| **Area** | **Percent of Grade** | **Academic Honesty Notes** |
|---|---|---|
| Activities | 30% (approximately 6 activities worth 5% each) | ALL work is your own, including programs, text, and images. Where outside help is received, you should acknowledge that help. |
| Projects | 30% (three projects of 10% each) | ALL work is by your group, including programs, text, and images. Where outside help is received, you should acknowledge that help. |
| Final Projects | 30% (one project) | |
| Participation and Attendance | 5% | This grade is for participation in group projects and daily attendance. |
| Putting away kits | 5% | Your kit needs to be returned exactly how it was received by you by the end of the exam period; no exceptions. |

All grades will be available on Canvas. Assessments are intended to determine student understanding of the course content. Student participation and effort are both taken into account for labs and the final project. Grades for students working in teams will reflect each student's contribution to the team.

The School-wide Policy on Late and Incomplete Work will be followed, including reporting late work. Assignments submitted after the due date and time are subject to a late penalty of 10% until the instructor completes grading the assignment, after that time the penalty will be 20% until 1 week before the next interim or final grading period, after which the grade will be zero for that assignment. The instructor will mark missing assignments as zeros in Canvas after the assignment has been graded, but the student can submit the assignment for a reduced grade as indicated above up until a week before the next grading period.

The instructor may also accept re-submissions for re-grading unsatisfactory assignments if the student has shown a good faith effort to seek help and demonstrate their improved understanding of the concepts. Re-grading may be subject to up to a 10% penalty at the instructor's discretion. Re-submissions will not be accepted after 1 week before the end of the next grading period.

**Grading Scale**

| | | | |
|---|---|---|---|
| | 88-89 B+ | 78-79 C+ | <70 D |
| 93-100 A | 83-87 B | 73-77 C | |
| 90-92 A- | 80-82 B- | 70-72 C- | |

**Sample Weekly Schedule:**

| **Week** | **Content** | **Assessments** |
|---|---|---|
| Week 1 | • Getting Started: Class Introductions, Course Overview, and Expectations<br>• Introduction to Robotics<br>• Introduction to Programming<br>• Programming Environment Installation | |
| Week 2 | • Self-paced Python Basics Modules | |
| Week 3 | • Taskbot Robot Construction<br>• MicroPython Introduction | |
| Week 4 | • Sensors (Touch Sensor and Gyroscopic Sensor)<br>• Motors (Rotary Encoders) | Basic Motion Assignment |
| Week 5 | • While Loops and Timing | Timing Assignment |
| Week 6 | • Turning Concepts and Methods<br>• Ultrasonic Sensor | Turning and Detecting Assignment |
| Week 7 | • For Loops<br>• Color Sensor | Turning and Detecting Challenges |
| Week 8 | • If…Elif…Else Statements | Program Branching & Choice Assignment |
| Week 9 | • Using Functions to Optimize Code | Function Definition Assignment |

| | | |
|---|---|---|
| Week 10 | • Basketball Drill Group Project | B-ball Assignment, Performance, and Report |
| Week 11 | • Line Follow Group Project | Line Follow Assignment, Performance, and Report |
| Week 12 | • Pirate Bot Group Project | Pirate Bot Assignment, Performance, and Report |
| Week 13 | • Final Project Group Project<br>• Complex Multi-task Design Project | Final Project Plan |
| Week 14 | • Final Project Group Project<br>• Complex Multi-task Design Project | Final Project Performance and Final Report |
| Week 15 | | |