General Information:

Instructor: Darren Dawson ([ddarren@clemson.edu](mailto:ddarren@clemson.edu))
Office: Rigs 105
Office hours by appointment only
TAs: Nick Watts ([nwatts@clemson.edu](mailto:nwatts@clemson.edu))
Office: Fluor Daniel 352
Available during scheduled class times or by appointment
Location: Holtzendorff Project Lab
Meeting Time: TBD
Course Webpage: [http://www.clemson.edu/ces/departments/ece/undergrad/mindstormslib.html](http://www.clemson.edu/ces/departments/ece/undergrad/mindstormslib.html)
Credits: 1

Course Description:

In this ECE robotics project, student teams will learn engineering and design principles by building and programming robots with Lego Mindstorms and MathWorks MATLAB. The Mindstorms kits allow for easy construction of advanced robots complete with servo motors and a variety of sensors. Using MATLAB, a mathematical programming tool widely used in industry and academia, to control the robots opens up a vast feature set beyond that provided by the basic Mindstorms software. In the first part of the class, students will go through several preliminary experiments to learn the basics of controlling the robot’s motors and reading from its sensors with MATLAB. Those sensors include touch, light, color, and ultrasonic. Also, MATLAB and its built in image processing functionality makes possible the use of a standard webcam as an additional sensor. After gaining a firm understanding of the fundamentals of design and the use of the Mindstorms and MATLAB toolbox, these student teams will complete a design challenge by designing, building, and programming their own two-legged walking robot.

Attendance Policy:

The Holtzendorff project lab will be available during the scheduled class meeting time for students to work on projects. The TA will be there during those times to answer questions and provide assistance. Students can stop in at any time during those hours. Students will be also able to take the Mindstorms kits with them and work on the projects outside of class if they choose, so attendance is not required. However, students will have to come during one of the scheduled meetings each of the first five weeks to get their work on the lab for that week checked off.
### Schedule:

<table>
<thead>
<tr>
<th>Item</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekly progress &amp; individual participation</td>
<td>Semester long</td>
</tr>
<tr>
<td>Introduction to class</td>
<td>Week of 1/17</td>
</tr>
<tr>
<td>Sample robot construction &amp; Lab 1 (Connections)</td>
<td>Week of 1/17</td>
</tr>
<tr>
<td>Lab 2 (Motors)</td>
<td>Week of 1/24</td>
</tr>
<tr>
<td>Lab 3 (Touch Sensor)</td>
<td>Week of 1/31</td>
</tr>
<tr>
<td>Lab 4 (Color &amp; Light Sensors)</td>
<td>Week of 2/7</td>
</tr>
<tr>
<td>Lab 5 (Ultrasonic Sensor)</td>
<td>Week of 2/14</td>
</tr>
<tr>
<td>Introduction to final project</td>
<td>Week of 2/14</td>
</tr>
<tr>
<td>Robot Prototype &amp; Progress Report</td>
<td>Week of 3/21</td>
</tr>
<tr>
<td>Finished robot &amp; Final Report/Presentation</td>
<td>Week of 4/25</td>
</tr>
</tbody>
</table>

### Grading:

Completion of all assignments is required. Grades will be heavily based on participation and effort put forth in each part of the class. Since attendance is not required, the two reports and presentation are the place to demonstrate effort. Reports and presentations that clearly describe the students’ methods for working through and solving problems will be scored high.

The prototype should demonstrate progress towards the final goal. Each group should have finished building their robot and should have begun the programming phase. The progress report should be no more than 1 page and the final report no more than 2. Both should include specific examples of challenges met and how they were overcome.

The final presentation will be submitted by email. Instructions will be given for recording narration in PowerPoint. The presentation must also include a video of the robot in action. It should be approximately 5 minutes long and cover much the same material as the reports.

### Academic Integrity

Clemson University's academic integrity policy reads: “As members of the Clemson University community, we have inherited Thomas Green Clemson's vision of this institution as a "high seminary of learning." Fundamental to this vision is a mutual commitment to truthfulness, honor, and responsibility, without which we cannot earn the trust and respect of others. Furthermore, we recognize that academic dishonesty detracts from the value of a Clemson degree. Therefore, we shall not tolerate lying, cheating, or stealing in any form.” Students will be held to this standard.
Welcome to ENGR 190

Mindstorms Meets Matlab
Solving the Rubik’s Cube

For the video, see the course website

http://people.clemson.edu/~nwatts/mindstorms/projects
General Information

Class Time: TBA
Holtzendorff Project Lab

Instructor: Darren Dawson
ddarren@clemson.edu
Office: Riggs 105
Office hours by appointment only

TA: Nick Watts
nwatts@g.clemson.edu
Office: Fluor Daniel 352
Office hours by appointment
Course Webpage

http://www.clemson.edu/ces/departments/ece/undergrad/mindstorms/lab.html

- Syllabus
- Lab manuals
- Sample projects
Class Schedule

• Week 1: Introduction
• Weeks 2-6: Labs
• Weeks 7-14: Design project
• Week 15: Final presentations
# Class Schedule

<table>
<thead>
<tr>
<th>Item</th>
<th>Due Date</th>
<th>% Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekly progress &amp; individual participation</td>
<td>Semester long</td>
<td>10</td>
</tr>
<tr>
<td>Introduction to class</td>
<td>Week of 1/17</td>
<td>-</td>
</tr>
<tr>
<td>Sample robot construction &amp; Lab 1 (Connections)</td>
<td>Week of 1/17</td>
<td>5</td>
</tr>
<tr>
<td>Lab 2 (Motors)</td>
<td>Week of 1/24</td>
<td>5</td>
</tr>
<tr>
<td>Lab 3 (Touch Sensor)</td>
<td>Week of 1/31</td>
<td>5</td>
</tr>
<tr>
<td>Lab 4 (Color &amp; Light Sensors)</td>
<td>Week of 2/7</td>
<td>5</td>
</tr>
<tr>
<td>Lab 5 (Ultrasonic Sensor)</td>
<td>Week of 2/14</td>
<td>5</td>
</tr>
<tr>
<td>Introduction to final project</td>
<td>Week of 2/14</td>
<td>-</td>
</tr>
<tr>
<td>Robot prototype &amp; progress report</td>
<td>Week of 3/21</td>
<td>20</td>
</tr>
<tr>
<td>Finished robot &amp; final report</td>
<td>Week of 4/25</td>
<td>20</td>
</tr>
<tr>
<td>Final presentation</td>
<td>Week of 4/25</td>
<td>25</td>
</tr>
</tbody>
</table>
Logistics

• Place: Holtzendorff Project Lab
• TA: Nick Watts
• Project Lab Schedule
  – To be determined
• Teams of Two Students
Design Project - Walker

- Design
- Build
- Program
- Details will be available later on the webpage

For a video, see http://robotics.benedettelli.com/ResonantBiped.htm
Record a PowerPoint Slide Show (Office 2010)

PowerPoint allows you to easily record timing and narration for a slide show. You simply click “Record Slide Show” and then start talking (assuming your computer has a microphone). The timing of when you advance slides will also be recorded.

Instructions

From the “Slide Show” menu, select “Record Slide Show.” You can choose to either start recording from the beginning of the presentation or at the current slide.

This should bring up the full screen slide show view along with a small window in the top left corner of the screen.

This window controls the recording. The arrow button on the left advances to the next slide, the pause button pauses recording, and the U turn discards the recording for the current slide and starts it over. The time on the left is the time recorded for the current slide and the time on the right is for the entire presentation.

When you are finished recording your presentation, close the recording window and exit the slide show. You can review what you’ve recorded by playing the slide show from the “Slide Show” toolbar.
You can also have your mouse pointer show up in the recorded slide show by holding both the CTRL key and the left mouse button while moving the mouse around. The pointer will be displayed as a “laser pointer.”

After recording a slide, in the left column of the “Normal” view, there will appear a star icon under the slide number and an audio icon in the bottom right corner of the slide to indicate that the slide has a transition and narration set.

The audio icon also appears in the bottom right corner of the slide itself in “Normal” view.

You can mouse over this to play the narration or adjust its volume. Also, you can select the icon and delete it to remove the narration and timing from that slide.
**Microphone Settings (Windows 7)**

To adjust the volume on your microphone, open the Control Panel. Click on “Hardware and Sound” and then click on “Sound.”

![Hardware and Sound](image)

In the window that opens, select the “Recording” tab.

![Sound](image)

Select your microphone device, and click the “Properties” button. In the window that opens, select the “Levels” tab. From here, you can adjust the microphone volume and boost.

![Microphone Properties](image)
Export to Video

After you have recorded all your narration, you can convert your PowerPoint presentation to a video. In PowerPoint, open the “File” menu and select “Save & Send.” Under “File Types,” select “Create a Video.” In the options on the right side of the screen, select “Internet & DVD” and “Use Recorded Timings and Narrations.” When you are ready to start, click “Create Video.” This process can take several minutes, depending on the length of your presentation and the size of embedded videos and narration.

You will be prompted for a location to save the video. This will create a Windows Media Video (.wmv) file, which is playable in Windows Media Player.
The following are examples of a good progress report, final report, and presentation from Fall 2010
Our assignment was to design, build, and program a Segway robot that would be able to balance itself and operate without outside interference or assistance. This project has presented us with many challenges, from testing and choosing a design to writing a correctly functioning program in MatLab.

We first started out with a design found on the internet called the NXTWay (cite) but soon found that the NXTWay design was not going to work for us. Next we decided to modify the design to try and lower the center of gravity and point of rotation of the axis by moving the wheels to the outside rather than underneath the robot and also moving the NXT down as low as we could put it. After continued testing we found that lowering the center of gravity made it so that our hardware and software could not successfully run in the manner that was needed to achieve balance and stability. So, we then decided to go with the Segway design found on a link from the LEGO Mindstorms webpage (cite), which consisted of a robot with a much higher center of gravity and an attached “rider”. This “rider” helped to balance the vehicle and could possibly be mechanized at a future point once the balance programs had been finalized. We have also decided to add a second Light Sensor to help with obtaining accurate readings. After running tests on the robot using both light sensors we determined that in order to make a more precise reading from the second sensor we would have to modify the design again to allow for a lower mounting point for the second light sensor (Note: both light sensors are mounted on the bottom of the robot, one in the front and one in the rear of the chassis). And now both of the sensors are at approximately the same height from the surface. This allows for more consistent readings. Since we have had the best results by far we have decided to stick with this design for the remainder of the project.

Our design was not the only thing to undergo many changes. We have constantly been changing and rewriting our program for the robot. Due to the constantly fluctuating light in the project lab we have to continuously change our “target” for the light readings and then fix the other variables that depend on the “target”. Also, we have opted to use a free whiteboard as our test sheet because it is smooth and one continuous color, unlike the test pad. A significant change to the program occurred when we decided to use “direct motor command” instead of “NXTMotor” because it offers a faster response and quicker activation of the motors. We have also replaced the “GetLight” commands in the program loop with a set variable “current” so that the light sensor will not a return more than one reading per loop. This allows the robot to act more accurately and with less confusion. Also, upon adding the second light sensor we have began using more complex commands such as “If-Or” instead of just “If” statements, which allows for more flexibility and for the reading of multiple sensors at one time.

We have not fixed all of the bugs, but the problems yet to be solved are few though important. A difficult challenge yet to be overcome is finding the balance between reactivity and power. We’ve written programs that have the necessary fast reaction time, but lack the power to keep the robot upright. We have also written programs that allow for the necessary power for the corrections, but it does not have the quick reaction time. To solve this we are working on splicing these two programs together, though so far we have been unsuccessful.
WORKS CITED

DESIGN 1:
NXTWay \( \rightarrow \) [http://philohome.com/nxtway/nxtway.htm](http://philohome.com/nxtway/nxtway.htm)

DESIGN 2:
Segway w/ Rider \( \rightarrow \) [http://www.nxtprograms.com/NXT2/segway/steps.html](http://www.nxtprograms.com/NXT2/segway/steps.html)
Segway Robot Final Report

The design of the SEGWAY robot is fairly simple. The motors were attached to either side of the robot using long blue connector pieces and wired to the NXT in MOTOR_A and MOTOR_C. The light sensor was then attached to the bottom behind the SEGWAY using a small LEGO_Technic piece that allowed the sensor to point vertically downward. We chose the light sensor because it is more sensitive to changes in light than the color sensor and the ultrasonic sensor is probably the least accurate sensor of all three since its smallest denominator is a centimeter.

The challenge of the design was how to make it easier to balance. The initial design involved many pieces in the kit and turned out to be quite bulky. The balance point became very far off center and impractical to balance. Later we decided that simplicity was best and simply attached the wheels to the controller and set the light sensor in front. We used as little pieces as possible and it worked for our purposes.

After concluding the project we came to the realization that there could have been a slight workaround to the high latency of the robot (a topic discussed later in the paper). If we had built the robot to be tall, thin, and have a center of mass near its middle then it would have tilted much slower than with our simple design which had a center of mass near the wheels. This would have given the robot time to realize it was not balanced and adjusted itself before it
was too late. This is a working theory but in essence it should have helped. Maybe it wouldn’t have done enough to compensate, but there should have been a noticeable difference.

The labs used to help us learn how to program the robot using MATLAB were very useful. In fact, we used a similar script from one of the labs to program the robot for our final assessment. We enjoyed the hands off approach to this creative inquiry where we were given assignments with the instruction included. This allowed us to work at our own pace which was probably much quicker than the other students during the project, and this allowed me to balance my classes effectively without rushing to complete any work.

The programming used for the SEGWAY script was done through MATLAB as instructed using the motor control kit supplied. We decided on a script that allowed the SEGWAY to alter its movement based on the light reading given by the touch sensor. The NXT is started at its equilibrium point of balance and the script is run. Once ran, the light sensor registers this point as the target level of brightness to maintain. If the light is brighter than the target level, it moves forward to compensate. If it is darker than the target level it moves backward. This way the NXT is always upright and hopefully staying in place where it should be.

One of the problems associated with this project was lighting. Although the light sensor is the best sensor for the job, it requires no alterations in the environmental lighting. Since the sensor is fixed to the front of the SEGWAY and not free to move, the rate at which light increases or decreases when the SEGWAY tilts is not constant. As a result, the movements of the SEGWAY are very jerky. This did not seem to make the Segway impossible though. We
simply adjusted the programming to have a maximum power output of -100 and a minimum of 100 which was more than enough power to correct its offset balance.

The response time and lighting was the major issue in this project. Even connected to the computer through USB the response was still too slow and the NXT is too far off balance before it realizes it is off of its center. This could have been worked around if we had a supply of larger wheels to raise the center of rotation but we did not have these at our disposal. We also attempted to program the Segway using DirectMotorCommand by TA’s suggestion as it reduced the latency between computer and machine. This had no visible effect on the robot’s poor performance as the difference between DirectMotorCommand and our original script was negligible.

In the end the Segway was limited to balancing with the need of outside assistance. By keeping a finger ready to provide some resistance when the robot began to fall, it was given enough time to react and subsequently corrected its positioning. This goes further to conclude that latency was indeed the reason the robot could not stand on its own two wheels. There are videos online of other robots built in the same fashion balancing on their own, but they ran off of code downloaded onto the robot, which resulted in zero latency. Some of them also had a gyroscope sensor, which is a much more accurate device for gauging displacement from the balance point.

We learned through this project how to use MATLAB, logical processes necessary to overcome some obstacles in an engineering setting, and simple design procedures. MATLAB will be useful in the future to me as an aspiring electrical engineer as it is a universal programming
language used on the field as well as in my future Clemson classes. Logic is also essential to an engineer, because if you couldn’t proceed to analyze a problem and be able to plan the steps to fix it you’d be in huge trouble. Since we both have a background of building toys in our day this project was a fun way to incorporate the process of experimental design for a Segway that had a practical application in the field of schematics or plans. Overall we were dissatisfied that our Segway was limited to balancing with assistance, but we felt that this project was entertaining and we learned a lot from the design process. We wish to continue with this creative inquiry next semester in building the Rubric’s Cube.
LEGO SEGWAY
A Production of Engr 190 Students
The intent of the design is to acquire and compose information about the mechanical, electrical, and computer science of Lego Mindstorms and Matlab. Once the information has been acquired the idea was to create a miniature Segway with Lego components and sensors.
Background Information

- **Segway** - (trademark) a self-balancing personal transportation device with two wheels; can operate in any level pedestrian environment
Learn and tested simple designs using

- **Motors**
  - Program directionality and speed
- **Touch**
  - Determined if something was pressing the sensor. When sensor was pressed, motor command changed.
- **Color**
  - Lead to line following programs, color identification
- **Light**
  - Line following
- **Ultrasonic sensors**
  - Used to move around obstacles, measure distances
Qualifications and Pre Design Process

- Learn and tested simple designs using
  - Motors
    - Program directionality and speed
  - Touch
    - Determined if something was pressing the sensor. When sensor was pressed, motor command changed.
  - Color
    - Lead to line following programs, color identification
  - Light
    - Line following
  - Ultrasonic sensors
    - Used to move around obstacles, measure distances
Learn and tested simple designs using

- Motors
  - Program directionality and speed
- Touch
  - Determined if something was pressing the sensor. When sensor was pressed, motor command changed.
- Color
  - Lead to line following programs, color identification
- Light
  - Line following
- Ultrasonic sensors
  - Used to move around obstacles, measure distances
Mechanical development

1st design
- Primary designed to be balanced similar to the Segway. It had a horizontal balance. The brick was length wise perpendicular to the ground (upright)

2nd design
- Focused on lowering the center of mass, balanced horizontal. Lower the center of mass allows better balance. The brick was parallel to the ground
- Problem: Center of mass was not lowered only distributed over axis

3rd design
- Lower the center of mass under the top of the motors. The brick was lowered vertically.
Electrical development

The initial reading was formed by taking the average of 8 values

- Normal Controller
  - The purpose of this controller was to correct the direction of the motor based on the comparison of the current height to the original height
  - Problem: Delay between the computer and the motors
  - Solutions: Direct Motor Control
  - Problem: Correction amount was not appropriate (Jerky)

- PID Controller
  - Correct the motion of the robot in a smooth manner because the derivative takes into account the error of rotation and the integral tries to fix the rate of spin and reduce the amount of error
  - Problem: Delay between computer and brick output
Programming in Matlab was a concept in this course. Coming from a minimal programming background, it was insightful to learn a new language.

Time management was a crucial skill gained from this project. Scheduling time to work and meeting deadlines was an important component in the segway production.

The method of production was important to understanding the project. We learned the process of visualizing an idea, to physically creating the idea. We furthered the process by taking the mechanical aspect and creating code for it. Also the ability to be able to translate from a real world concept into the code of the mechanics behind it was crucial to this project.
The robot did not function as expected.
Non of the ideas fully functioned as they should.
The ideas that worked the best involved the lowest center of mass and functioned the best under the conditions of our program (PID controller).
Bloopers
LEGO SEGWAY

A Production of Engr 190 Students
The Design Process

- **Mechanical development**
  - 1\textsuperscript{st} design
    - Primary designed to be balanced similar to the Segway. It had a horizontal balance. The brick was length wise perpendicular to the ground (upright)
  - 2\textsuperscript{nd} design
    - Focused on lowering the center of mass, balanced horizontal. Lower the center of mass allows better balance. The brick was parallel to the ground
    - Problem: Center of mass was not lowered only distributed over axis
  - 3\textsuperscript{rd} design
    - Lower the center of mass under the top of the motors. The brick was lowered vertically.