EEL 4932: Autonomous Vehicle Design
Spring 2002

Classroom:	MW 5:00-6:15, ENGR 227
Instructor:	Dr. Fernando Gonzalez
Office:	ENGR 444
Office hours:	Mondays and Wednesdays 2:30 to 4:30, Thursday 3:30 to 5:30
Phone:	823-3987
Email:	fgonzale@pegasus.cc.ucf.edu
Web:	http://pegasus.cc.ucf.edu/~fgonzale/

References: Saeed B. Niku, "Introduction to Robotics: Analysis, Systems, Applications", Prentice Hall, 2001. Handouts provided during the course.

Prerequisites:

MMAE	Modeling Methods (PR), Measurements (PR)
EE	Electronics I (PR)
CpE	CSD1 (PR)

Goals: This multidisciplinary course is designed to give the students a variety of basic concepts and hands on experience in robotics and automation. Topics include control, sensing, vision, intelligence, and mechanics. The course will include a series of lectures from the course instructor as well as from other faculty and students who are experts in their respective studies. In order to gain hands on experience, the students will participate in a project in which they will design and build an autonomous vehicle that will participate in an international robotics competition.

Competition: 10th Annual Intelligent Ground Vehicle Competition

Topics: Each of these topics could be a course within itself; however, this course will only introduce them:

- Robotic controls, including an introduction to robotic kinematics and dynamics
- Data acquisition from GPS and other sensors
- Video sensing, image capture, robotic vision, and basic object recognition
- Artificial intelligence, robotic motion planning
- Electric motors, variable speed motor controllers

Grading:

January 23	**Proposal** (brief documentation and open-floor presentation; 10 minute time limit):	**20%**
	Responsibility to teach other students how your project works	
	Presentation of alternatives	
	Presentation of chosen method	
	Purchasing information	
February 18	**Design** (documentation and presentation; 10 minute time limit):	**30%**
	Final design selection:	
	Algorithms (pseudo code)	
	Hardware	
	Schematics	
	Specification Sheets	
	Purchase orders	
April 15	**Final Presentation** (deliverables and presentation):	**40%**
	Participation	**10%**
	Class attendance	
	5-minute weekly meetings	

Proposal
The documentation is aimed at establishing the exact task and its specifications. It must include the following:

- Your selected group, the selected project and the individual project breakdown between the members in the group.
- The design method that will be used to implement the project. For example, what type of algorithm, what type of circuit, exact pseudo code or diagrams is not necessary at this point.
- The deliverables include its functionality.
- Breakdown of material needed to be purchased with their corresponding price.
- Approximate time table

The presentation is aimed at educating the class about your project so that we all know what every one is doing. The presentation is limited to 10 minutes. It must include the following:

- A brief, easy to understand explanation of your project without insignificant detail.
- A presentation of the alternative methods you considered before selecting the one to be used.
- A presentation of your chosen method.

Design
The documentation is aimed at establishing a set of design specifications including all known details. It must include the following:

- All algorithms that will be used. You may use pseudo code, charts and other diagrams.
- All circuit diagrams that will be used.
- A list of all hardware devices that will be used.
- The specification sheets of the hardware devices. If the specification sheets are too large then summarize it and indicate where to find the specification sheets.
- Purchase orders.

The presentation is aimed at educating the class about your selected methods so that the class may cretique it or give you ideas. The presentation is limited to 10 minutes and must include the following:

- Brief explanation of the algorithms.
- Brief explanation of the circuit schematics
- The list of hardware to be used and what each does.

Final Presentation
The documentation is aimed at recording your work so that it may be modified or reproduced by other people in the future. It must include the following:

- All algorithms that were used. You may use pseudo code, charts and other diagrams.
- Code listing with good documentation in the code.
- A structured chart, or other types of charts that you deem appropriate, to present the structure of your code, i.e. what function calls what function and what information is passed.
- Diagrams explaining all the data structures used.
- Results of software testing. This proves it works.
- All circuit diagrams used.
- Plots of any circuit output like the information in data books. Only include data that is relevant.
- Building maps of the circuits. This allows other people to modify the circuits.
- A list of all hardware devices used.
- Any significant characteristic about the hardware that was not obvious from its specifications. This will save time when other people try to work with that hardware.
- The specification sheets of the hardware devices. If the specification sheets are too large then summarize it and indicate where to find the specification sheets.
- A complete list of all material including all hardware as well as wire, chips, etc that was used.
- An approximate number of hours spent on each task. This gives an idea to other people of the difficulty.

The presentation is aimed at demonstrating to the class you accomplishments during the course and must include a demonstration of the functionality of your project. This may be a film if the actual demonstration through execution is not feasible. The presentation is limited to 10 minutes.

Code Documentation
All software code must be well documented. The documentation must be in the code itself using the /* … */ or the C++'s // *comment to end of line* features. Each function must include the following information:

- What the function does.
- The algorithm it uses.
- The input.
- The output.
- Any assumptions.

For example:

```
// This function pushes an integer onto the stack and updates the stack's front pointer.
//
// Algo:
//      Get a new node.
//      If you got it successfully
//              Copy the inputted integer to the data field.
//              Copy the FRONT pointer to its NEXT field.
//              Copy the address of this new node to the FRONT pointer.
//              return 0
//      else
//              return –1
//
// The inputs are:
//      DATA   -        The integer to push.
//      FRONT -         The pointer to the front of the stack.
//
// The outputs are:
//      none.
//
//Assumptions:
//      If a new node is not created the function does not print an error message.

int  push( int data, node *front)
        {
        code.
        }
```

Remember that you are working with a large team. This is not the situation you have in most other classes where the code will only be looked at by you and will never be modified after the semester end. In this course we all will need to understand every ones code. Also this is not an academic exercise that you trash at the end of the semester. Other people in future competitions must understand and modify this code. Functioning code that is not well documented will be useless in any future competitions. Also this code is to be delivered to our sponsors as part of an agreement for there support.

Circuit Diagrams
All circuit diagrams must be such that I can reproduce the circuit board. It must list all parts numbers used. I must also be able to look at the circuit diagrams and the circuit boards and figure out where each point in the diagram is on the board. Other people in future competition will be interfacing and modifying the boards. It is extremely important that the circuit diagrams are accurate and up to date. You must maintain your diagrams up to date with and boards. If you do not maintain the diagrams up to date I will have you disassemble the board, determine the diagram and rebuild a new board!!