BMW: Brainwave Manipulated Wagon

Zijian Chen, CSE, Tiffany Jao, CSE, Man Qin, EE, and Xueling Zhao, EE

Abstract —**BMW (Brainwave Manipulated Wagon) is a robotic car that can be remotely controlled using the user's brain signals. This system uses BCI (Brainwave Computer Interface) to provide communication between our brain, the computer application and the robotic car. It uses a commercial EEG headset to acquire EEG data, classifies and interprets the data set on the computer application, and achieves desired commands on the robotic car based on the classification. The purpose of the BMW is to demonstrates the feasibility of applying in BCI helping people with physical disabilities, as will be shown with the demonstration of controlling the robotic car.**

I. Introduction

People with motor disabilities are limited in the physical activities they can perform in daily life due to their constraints, such as not being able to walk or move independently. In the US, physically disabled individuals make up the largest minority group of the US, and about 74.6 million people have some kinds of physical disability [1][2]. Some existent solutions are present to solve the problems of physical limitations, which include implementing prosthetic limbs, using wheelchairs, or hiring someone to take care of them. These solutions have solved part of the problems and allowed the people in the group to live normal lives. For those who lose their limbs, they are able to implement the prosthetic limbs or walk with crutches so that they can move freely. However, for people who could not move their hands or feet flexibly (e.g: paralyzed), they even have a hard time controlling their surroundings by themselves using a wheelchair. Therefore, a regular wheelchair could not fulfill their needs. At this point, a better solution would be the BCI (Brainwave Computer Interface).

BCI allows direct communication between brainwaves and the external world. With BCI, non-muscular communication and control is no longer speculation [3]. The person's messages and commands are not expressed by external control such as pushing a button; they are controlled by electrophysiological phenomena such as MEG (Megnetoencephalography) and EEG (Electroencephalography) features. EEG measures the voltage potentials generated by neural currents. It reveals the information of our brains. Over the past two decades, many studies have evaluated the possibility that brain signals recorded from the scalp could provide new technology that does not require muscle control [4]. The BCI solution is beneficial for the individual who is paralyzed without brain issues. The patients communicate with the external device , such as a wheelchair, with their brains, which is a huge improvement in their lives. And the problem in society about this specific group could be largely resolved because they are now more like normal people who could control their movements freely.

In our project, we are not going to build a complete wheelchair using BCI; instead, we are building a robotic car for demonstration because of time limitations. A real BCI controlled wheelchair takes time to build up and implement. If the robotic car is implemented successfully, the functionalities are applicable to a real wheelchair system. There are a few specifications of the robotic car. The robotic car is associated with a Neurosky EEG headset for control. It requires basic movements such as moving forward, stopping, and turning, just like a real car does. The remote distance should be at least within 30m, as we would only need to control the car within eye sight. The battery life is required to be three hours or longer. This is a reasonable time endurance as it doesn't need to be too long. The robotic car should be compatible for every user. Software is developed to help training, and the training set data is collected for analyzing. Here is our specification table:

TABLE I
Specifications

Specification	Value
Car Direction	Forward/Stop/Turn
Remote distance	30 m
Battery Life	<3 hours
Compatibility	Every user
Command Accuracy	75%

II. Design

A. Overview

To build a BCI controlled robotic car, we will use a commercial headset to retrieve the EEG data and send it to a computer software application. On the back-end of the computer application, the Fast Fourier Transform (FFT) will be performed on the acquired EEG data, which will be used by the command algorithm to classify different commands. On the front-end, a user interface will direct the user to perform the training and control process. To communicate with the robotic car, the computer will send the classified command as digital data through the serial port to a transmitter(TX) module. The transmitter (TX) module will then transmit the signal to the receiver (RX) module located on the robotic car. Both the TX and RX module will consist of a microcontroller and a wireless radio, more specifically, Arduino Uno and XBee radio.

We chose to use the Neurosky Mindwave [4] headset as the solution to obtain the EEG signal. The reasons for choosing

T. Jao from Longmeadow, Ma (e-mail: tjao@umass.edu).
X. Zhao from Quincy, Ma (e-mail: xueling@umass.edu).
Z. Chen from Malden, Ma (e-mail: zijian@umass.edu).
Q. Man from Wuhan, China(e-mail: mqin@umass.edu).

this headset are primarily due to its affordable price, its software development kit(SDK) resource, and the amount of BCI research using the Neurosky headset that is available for study. Having the BCI controlled using Neurosky usually takes advantage of the attention and mediation values, computed by Neurosky's proprietary eSense™ algorithm. However, Neurosky only contains a dry EEG electrode placed at FP1 location, which is located on the forehead above the left eye[4]. Since we can only retrieve EEG data from one location, more specifically, the frontal lobe of the brain, it limits the number of commands that we can distinguish with the EEG data, as a typical BCI system generally takes the signal from different channels and classifies it[5]. However, in the research paper *Mental Task Classification Using Single-Electrode Brain Computer Interfaces* [6], the Neurosky Mindset headset was used to attempt to achieve the classification of three tasks, with an average accuracy of 77.6 % using a Bayesian classifier [6].

Alternative methods for retrieving EEG data for BCI control include using another headset, Emotiv [7] and an EEG amplifier. Emotiv was one of our choices, as it includes more EEG sensors located in different region of the brain, but the price of the EEG version of the headset is above the given budget, making it unaffordable for us. An EEG amplifier and active electrode system was also considered, but the setup process requires correct placement of the electrode, which was complicated for different users to set-up a similar system in previous experiments.

Also, we decided to design a software application on the computer, instead of using an embedded solution or directly connecting with the robotic car for two reasons. First, the headset uses Bluetooth v3.0 protocol, which supports remote distances up to 10m [8]. Thus, a direct connection between the robotic car and the headset is undesirable, as it limits the remote distance connection. Second, a training set is required for BCI, as listed in the feature properties in *A Review of Classification Algorithms for EEG-based Brain–Computer Interface* [5]. We need to decide the stimuli that we can use for this particular headset, and a computer application is an appropriate choice for setting up a standard experiment.

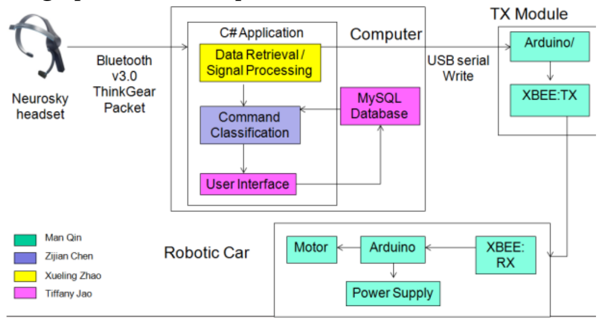


Fig.1: System block diagram

Fig. 1 illustrates our system block diagram, which includes several main blocks.

1. Data Retrieval/ Signal Processing:
   This block is responsible for retrieving EEG data from the Neurosky Mindwave headset, which uses the Bluetooth v3.0 protocol. The Fast Fourier Transform (FFT) is performed on the acquired EEG data for analysis.
2. Command Classification Algorithm:
   This block is responsible for utilizing the retrieved EEG data and classifying different commands. The criteria for classification are based on the collected training data sets from the database.
3. Graphical User Interface (GUI):
   This block will allow user to start the control, while providing sets of standard stimuli to display and help the user with the control.
4. TX(Transmitter) Module:
   This block consists of an Arduino Uno and a XBEE radio. The Arduino Uno will receive the digital signal of the resulting command from the software. It will then send this signal to the receiver module on the car.
5. RX(Receiver) Module/ Robotic Car:
   The RX(Receiver) Module consists of an Arduino Uno and an XBee radio. The RX module will receive the signal from the TX module. The RX module will send the received signal to the Arduino, which will control the robotic car based on the resulting command.

B. Data Retrieval

Data retrieval is the first task as it acquires EEG data from the Neurosky MindWave headset. The developer provides a library called Thinkgear for the users to utilize [9]. This library obtains functions that allow users to extract values from the headset directly for further analysis. Raw EEG data from the library is in terms of voltage. The sample rate of the headset is set to 512 samples per second, which means that every raw data point is outputted for every 2ms. Raw EEG data is important since it allows us to perform the FFT (Fast Fourier Transform). Other values that are useful in our project include attention level, meditation level, and Alpha/Beta power spectrum. Attention and Meditation values are presented as numbers ranging from 1 to 100. For example, attention level measures your concentration, and the number increases as users try to focus more. Attention is the main parameter for the project so far to achieve binary commands of LED light at MDR.

C. Command Classification Algorithm

This block is responsible for classifying commands for the robotic car based on the incoming EEG data, such as alpha and beta band power spectrums. Different commands are associated with the corresponding tasks that the user are directed to do. However, to achieve binary command for MDR, we use the attention value that Neurosky headset computed. According to Neurosky's user guide, the Attention value can be controlled through a visual focus, such staring at a point on the screen, or focusing on imagining the action user hoped to accomplish [10].

Currently, we chose to use only attention level in our algorithm, because we are still in the process of discovering the tasks and the association with band power that we can classify with one sensor. Also, attention level did reflect the concentration of users as we tested on different people. In addition, the implementation would still be similar after the associated wave band power is defined.

For MDR, since we are working toward distinguishing two states with attention values, we determined the classifying point between the two states, concentrated and not concentrated, as illustrated in Figure 2.

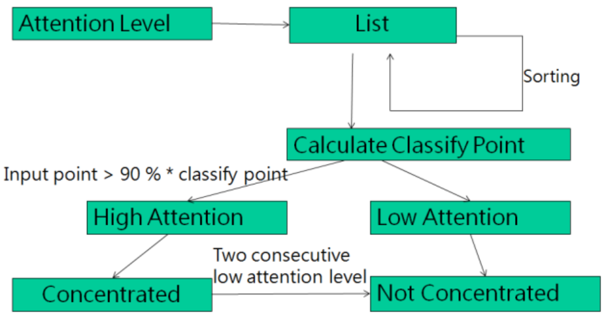


Figure 2: Command classification algorithm in MDR for two states

First, the training data set consists of a set of attention values collected while the user is concentrating and staring at the blinking light on the screen. Our algorithm sorts those data using merge sort, since merge sort has a faster runtime and does not affect the performance of our system. Then, we calculate the classifying point by averaging the middle third of the training data. This approach eliminates the outlier, such as the situation when user is not fully accomplishing the task during the training session. However, while we are testing it, we have discovered that using only the threshold to distinguish between two states may not yield a stable result. For instance, sometimes the test subject has a hard time sustaining the concentrated state in long intervals, even though the subject is still staring at the blinking light.

To stabilize the classifying result, we decided to include two other criteria. The first criteria is that, the algorithm will only classify the attention level as high when the input attention value is larger than 90% of the classifying point. In this case, input attention value will still be considered as high, even if it drops below the classifying point by a little bit. Otherwise, the input attention value will be considered as low attention level. The second criteria is that, if the user is at the concentrated state originally, two consecutive low attention inputs will change the state from concentrated to not concentrated. This criteria address the problem of when the user only has a short instance when they are unable to sustain the concentrated state, but they are still trying to do so and shortly recover after that instance.

During the experimental process with this algorithm, several observations were noted. If the user is fully committed to a single thought and is staring at the LED light intensively during the training process, the computed classifying point will be really high, and thus makes it difficult to go into the concentrated state during the control process. Also, if the user is in the concentrated state and is trying not to concentrate, avoiding staring at the blinking light on the screen, it will take at least two seconds to change the state because the algorithm requires two low attention levels to be considered as not concentrated.

To look at the possibility for having multiple commands, it is necessary for us to look beyond attention level and discover the correlation between band power and different external or internal stimuli. That is: we are trying to utilize Event Related Potentials(ERP), which is a phenomenon where exposures to external and internal stimuli could generate responses in an EEG wave [11]. Discovering the appropriate stimuli that can be used to stimulate an ERP response at the frontal lobe region is required. To determine the stimuli to test on, it is required for us to look at research in neuroscience, to understand the frontal lobe region, and what the past BCI control has been using.

To distinguish the band power pattern of using each stimulus, we will use Naive Bayes classifier [12]. This is a probabilistic classifier that utilizes the previous classification data to determine the current classification. The Naive Bayes classifier has several advantages. First, it has a linear run time, so it will not reduce the performance of our system significantly. Second, since it is a probabilistic classifier, more data will improve the accuracy of the classification. In this case, we will use a database to store all the band power data with different stimuli, and that data will serve as the input to the classifier. In addition, the Naive Bayes classifier will dynamically recompute a set of probabilities distribution when a new training data for specific stimulus is available. In this case, the user does not have control over the classification criteria.

However, we are currently facing some challenges regarding use of the Naive Bayes Classifier. In order to use it, with the sets of stimuli to test on, we still need to know what correlation to expect with one single EEG sensor, instead of relying on Bayes classifier entirely. Otherwise, if the Naive Bayes Classifier fails to classify the effect between two stimuli, it will be unknown to us whether the two stimuli are indistinguishable due to the fact that we only have one sensor or because we didn't have enough data. Furthermore, a large number of experiments and datasets will be needed to improve the accuracy of the classifier.

D. *Software Flow/User Interface*

The primary purpose of including a user interface block is to have a training interface that gather the user's training data with different stimuli, in order to achieve a better control after generalizing those patterns. The secondary purpose is to have a platform for a standard experiment to test the stimuli that could potentially work well with our goal. On the back-end, the software needs to handle the state change based on the user's operation, such as a button click. On the front end, the user interface is also responsible for displaying the selected stimuli to help user to achieve the control that they hope to accomplish. The output command is also displayed on the screen in order for the user to monitor the command being made.

To build the user interface, Visual C# was chosen as the development platform, as it has native graphical features for better visualization and is easy to put on different stimuli for testing. On the development side, C# includes a relatively easy interface for the developer to create different events, which is essential for the purpose of detecting the button click to trigger the program as well as setting up a fixed-time training interface with timer events.

To create a standard software flow, a preliminary state diagram is created to layout the necessary process that must happens within the software. Each state is dedicated to a certain task at this stage, but future state reduction and combination will be needed to eliminate the complicity of state transition. The state diagram design processes are adapted from ECE112, Introduction to ECE, and ECE373, Software Intensive Engineering. To speed up the data collection process, the use of multithreading is considered and will need to be learned to apply in C#. The concept of multithreading has been elaborated in Software Intensive Engineering as well.

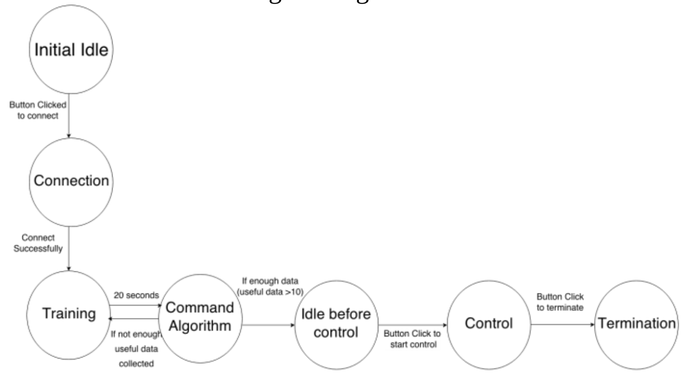
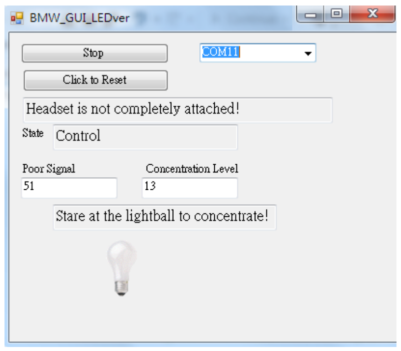

Initial Idle
Button Clicked to connect
Connection
Connect Successfully
Training
20 seconds
If not enough useful data collected
Command Algorithm
If enough data (useful data >10)
Idle before control
Button Click to start control
Control
Button Click to terminate
Termination

Figure 3: State diagram of the software

The current state diagram of the software is outlined in Figure 3. The current design for this stage is aimed for an experiment interface until we determine the usable stimuli for different commands. For now, basically, the program will start in the Initial Idle state, where the user would click the button to trigger the connection with the headset. If the connection is established successfully, it will then go onto the training state, where user will be directed to concentrate and stare at the stimuli for 20 seconds. After 20 seconds, the program will transition into the command algorithm state, which the collected data from the training state will be used to analyze. In the case when the headset does not attach to the head at a proper position, a new attention value will not be computed, and a poor signal value, computed by the headset, will indicate a non-zero number. When the poor signal value is non-zero during the training process, the collected training data is discarded. Then, if the command algorithm does not have enough training data to use, it will transition back to the training state, where the user will be asked to re-train and perform the same task. If the command algorithm does have enough training data, it will transition to the idle-before-control state, where the user could click the button again to start the control. During the control state, the user can click on the button to terminate the program.

Figure 4 illustrates the current GUI layout. The interface contains buttons for user to control the program. During the control state, it will inform user the output of the command algorithm, whether the user is classified as concentrated or not. It also displays some warning message. For instance, it will inform user if the user is not wearing the headset properly. It will also give indication of the state that the program is at.



Headset is not completely attached!

State: Control

Poor Signal: 51

Concentration Level: 13

Stare at the lightball to concentrate!

Figure 4. Current GUI layout shown on computer

E. *TX/RX and Robotic Car*

This subsystem is used to control our car with the command generated after command classification and consists of two parts. The first part is the Transmitter/Receiver pairs. They are required to be compatible with not only computers, but also the robotic car that we will build. They should also communicate with each other with sufficient speed and a long distance. The Arduino Uno and XBee were chosen to build this subsystem. We chose XBee as the wireless solution because it has the indoor range of 40 meters [13], which is sufficient for our needs. The Arduino Uno can also interact with XBee. We have four important steps to complete this subsystem. The first step is to transmit the signal from the computer to the Arduino. Then, we need to connect the Arduino and XBee and perform configuration. The third step is to establish the communication between the two XBees. The last step is to send the signal from Arduino Uno to the car.

We need to learn how to configure the XBee to successfully interface with the Arduino Uno. After build up all of the parts, we can easily test the subsystem by sending a signal from the computer and checking to see if there is the same signal measured after Arduino in the receiver terminal.

The other part is the robotic car, which will be designed for movements forward, backward, stopping, turning left or right, speeding up, slowing down and triggering power on or off, all while showing its status of action and having sufficient battery life. Since the receiver will get the digital signal, we will use an A/D converter to translate the digital signal to an analog signal, which will be used to drive our car directly after a power amplifier. The A/D converter is covered in Electrics I and the Power Amplifier is covered in Electrics II. Furthermore, we need a display to show what state our car is at. Finally, power is also an important constraint, so we will need to learn about the power system.

To test the whole block, we can send a chosen signal to check if the robotic is doing the expected operation and showing the correct state in the display. We can measure the power consumption in the system and then calculate the battery life.

F. *Signal Processing*

Signal processing is the sub part that establishes a connection between the headset and command algorithms in order to achieve controls. However, at the point of MDR, the result of signal processing has not yet utilized by the project itself.

With the present library available, we utilized the functions there to extract the raw EEG data and its corresponding attention level. The raw EEG data is assumed to be reliable. Noise filtering is not considered as important for the FFT analysis because the headset itself filtered out extraneous noise and electrical interference [9]. The dominant noise comes from the noise of muscle movement. This noise is detectable by observing the raw EEG data because it is very similar to the noise of eye blinking, which is an external function that is provided by the headset. If muscle movement noise is presented, obvious spikes will be observed from the raw EEG data. Therefore, the raw data extracted directly from the headset is reliable, and noise filtering could be ignored at this point. MATLAB is used as the main research tool so far to performing FFT analysis. C# language will replace MATLAB for FFT analysis because MATLAB is not favorable for our project development, as noted in our PDR presentation. The purpose of the FFT analysis is to look for the relationship between the brainwave signals and the attention level so that relationship is better defined. In order to achieve multiple controls, having the value of attention level is not enough. We focused on extracting the alpha wave power spectrum from FFT which is ranged from 8Hz to 12Hz. Figure 5 shows the distribution of the result.

Alpha Power Spectrum (FFT) vs Attention Level (headset)

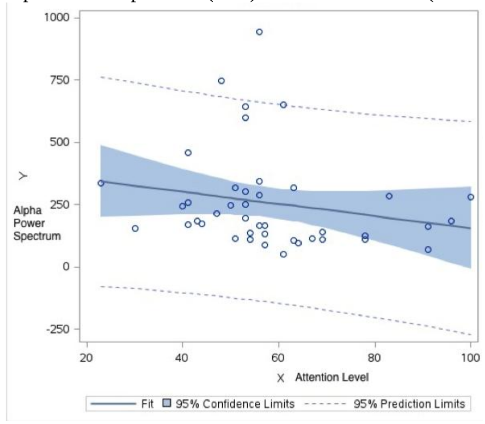


Figure 5. The graph is plotted using SAS (Static Analysis Software)[14] for analyzing. It shows a linear-like relationship between the alpha power and the attention level.

Figure 5 shows the relationship between the alpha wave power spectrum that is calculated from MATLAB and the attention level, that is computed by the headset itself. The units of alpha wave power spectrum are in watts, but the number is arbitrary due to a few conversions. However, it would not mess up the relationship because all points have the same scale. From the distribution, we could see that the alpha power spectrum has a linear-like relationship with respect with the attention level. The slope of the linear relationship is negative, which means that as the attention level increases, the alpha power spectrum decreases. However, the linear relationship is not too obvious. The scale of the power spectrum is in thousands, but the slope is only in tenths, which is too small to conclude the negative linear relationship. Therefore, we cannot make a solid conclusion at this point. That is the reason why signal processing has not yet been integrated with the command algorithm for controlling the LED. Also, one of the problems of the analysis is that the data set is taken without any environment setup. If multiple controls are desired, a well-defined experiment is required.

III. Project Management

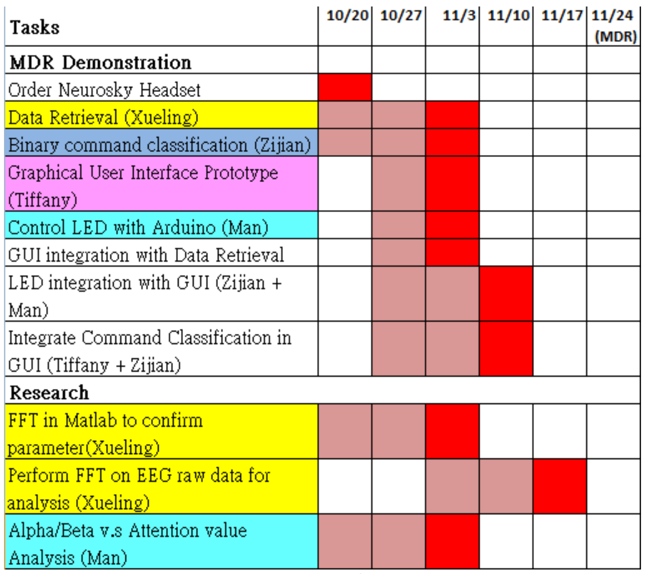
Table II below lists the MDR goal of our project.

Specification	Completion
Retrieval of EEG data from Neurosky	100%
GUI - Software Interface Prototype	100%
Determine Binary command from algorithm	100%
Demonstration of controlling LED on/off	100%

We have accomplished our MDR goals successfully. For MDR, we are mostly concerned with the task of achieving binary BCI control. Thus, we placed our focus entirely on the software and data analysis. We have demonstrated the connection between the Neurosky Mindwave headset, the computer, and the Arduino Uno. Our C# application is capable of displaying information to the user. On the back-end, the application can receive data from the Neurosky headset. The command algorithm can also classify binary commands, namely whether user is concentrated or not. In this case, we are directing the user to concentrate by staring at the blinking light on the GUI. The output of the classified command is sent to the Arduino board, which will control the LED light on or off.

To look beyond binary commands and attempt to control a robotic car, we are trying to analyze the relationship between the EEG band power spectrum and attention values. We have implemented the FFT in MATLAB, which allowed us to analyze the EEG band power data. It will be useful later as we define experiments to test with different stimuli.

Figure 6 illustrates our Gantt chart up to MDR.



Tasks	10/20	10/27	11/3	11/10	11/17	11/24 (MDR)
MDR Demonstration						
Order Neurosky Headset	🟥					
Data Retrieval (Xueling)	▨	▨	🟥			
Binary command classification (Zijian)	▨	▨	🟥			
Graphical User Interface Prototype (Tiffany)		▨	🟥			
Control LED with Arduino (Man)		▨	🟥			
GUI integration with Data Retrieval		▨	🟥			
LED integration with GUI (Zijian + Man)		▨	▨	🟥		
Integrate Command Classification in GUI (Tiffany + Zijian)		▨	▨	🟥		
Research						
FFT in Matlab to confirm parameter(Xueling)	▨	▨	🟥			
Perform FFT on EEG raw data for analysis (Xueling)			▨	▨	🟥	
Alpha/Beta v.s Attention value Analysis (Man)	▨	▨	🟥			

Figure 6: Gantt chart for MDR

The remaining part of our project includes building a robotic car that can be remotely controlled by the application. This will be done during winter break and the first two week of spring semester to ensure that we can integrate it before CDR. Furthermore, the software application also needs to setup a database for command classification to access, and the GUI needs to become more user-friendly.

Most importantly, we want to improve our command algorithm to achieve more controls using EEG waves. However, projects that we found using Neurosky headset can only achieve a maximum of three different commands using either its attention or meditation value and the detection of an eye-blink[6]. We are not optimistic about classifying different mental tasks to achieve multiple commands through the experience working with the headset, since we only have one sensor to work with. As an alternative, we can detect eye-blink using the Neurosky headset by observing an obvious spike on the raw wave data, as it is caused by the presence of muscle movement. It deviates from our goal to achieve controls utilizing only an EEG signal, which leads to our decision to switch to a EEG headset with more sensors, Emotiv [7], for CDR.

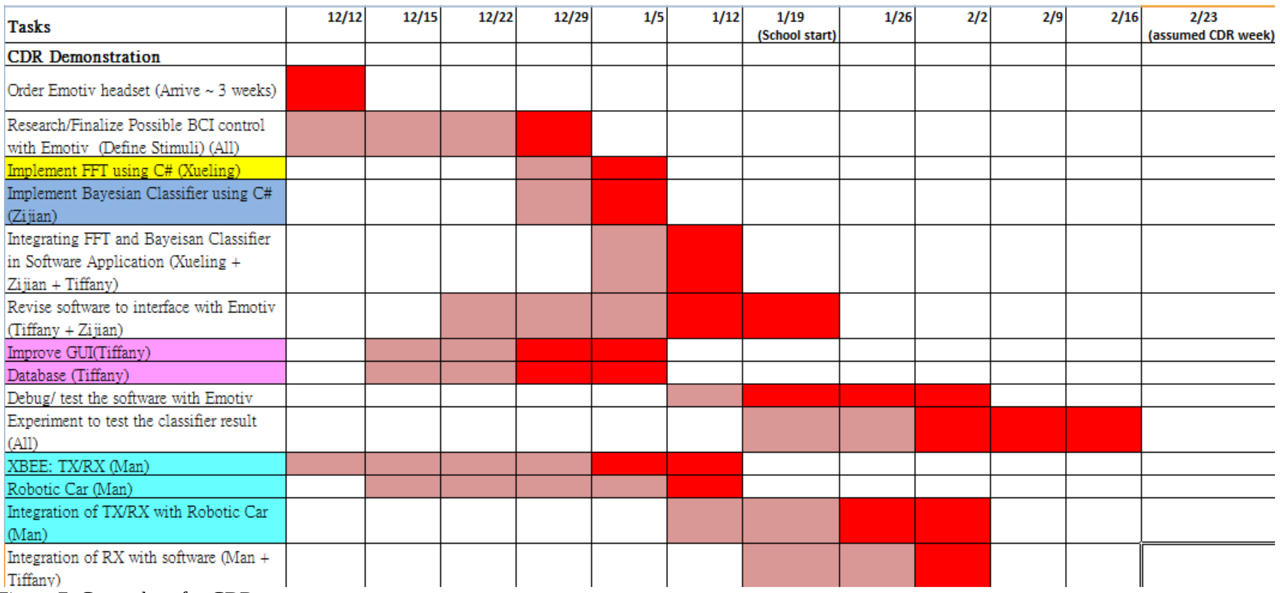
To make this plan feasible, we will start the software implementation without the Emotiv headset during winter break. With the Emotiv headset, the possibility of achieving multiple commands increases greatly. For instance, the SSVEP (Steady-state visual evoked potential) -based BCI system has been developed using Emotiv EPOC, where the system utilizes the blinking light at different frequencies as external stimuli to create corresponding ERP responses at the occipital lobe of the brain [15]. After we complete our research and defined the form of stimuli that we can utilize, we can then implement the FFT and Bayesian classifier in C# application. There is also a need to revise the GUI based on the stimuli that we decide to use.

Figure 7 is our Gantt chart for CDR.

Team Contributions

Each member in Team BMW made a significant contribution to the project. All members used their expertise to deliver valuable information to the team. Xueling is the EE of the group. She was responsible for data retrieval from the Neurosky headset to ensure that the connection was established successfully. She also worked on analyzing the FFT for raw EEG data and looking for relationship between the brainwaves and the attention level. Man is the other EE of the group, who is responsible for the hardware components. He was responsible for finding the relationship between the alpha and beta power spectrums, which are provided directly from the headset. He will be building a robotic car and working on the wireless communication between the robotic car and Arduino. Tiffany is the CSE of the group and the Webmaster. She was responsible for designing the graphic user interface and database for our application. Zijian is the CSE of the group and is a good software programmer. He was responsible for implementing the command algorithm and providing the technique support for other software parts of the project. He is also the team manager.

Each team member helps each other, and, in order to communicate with each other conveniently, we created an online chatting group to share our opinions at any time. We also used Google Docs to share our research information as well. Each team member is aware of the current progress of other members, and we all provide feedback to each other. Besides the online communication, we have at least one team meeting in person each week before our regular meeting with our advisor, Professor Xia. In our group meeting, we discuss our progress and collect feedback such as questions, challenges and opinions. Then, we start to integrate all the parts together. In the regular meeting with our advisor, we talk to Professor Xia about our project progress, and the questions and concerns that we have about the project.



Tasks	12/12	12/15	12/22	12/29	1/5	1/12	1/19 (School start)	1/26	2/2	2/9	2/16	2/23 (assumed CDR week)
CDR Demonstration												
Order Emotiv headset (Arrive ~ 3 weeks)	🟥											
Research/Finalize Possible BCI control with Emotiv (Define Stimuli) (All)	▨	▨	▨	🟥								
Implement FFT using C# (Xueling)				▨	🟥							
Implement Bayesian Classifier using C# (Zijian)				▨	🟥							
Integrating FFT and Bayeisan Classifier in Software Application (Xueling + Zijian + Tiffany)					▨	🟥						
Revise software to interface with Emotiv (Tiffany + Zijian)			▨	▨	▨	🟥	🟥					
Improve GUI(Tiffany)		▨	▨	🟥	🟥							
Database (Tiffany)		▨	▨	🟥	🟥							
Debug/ test the software with Emotiv						▨	🟥	🟥	🟥			
Experiment to test the classifier result (All)							▨	▨	🟥	🟥	🟥	
XBEE: TX/RX (Man)	▨	▨	▨	▨	🟥	🟥						
Robotic Car (Man)		▨	▨	▨	▨	🟥						
Integration of TX/RX with Robotic Car (Man)						▨	▨	🟥	🟥			
Integration of RX with software (Man + Tiffany)							▨	▨	🟥			

Figure 7: Gantt chart for CDR

IV. Conclusion

In this first stage, we built a simple system to test our control using an LED light. As a result, we have controlled the LED light to turn on or off with brainwaves successfully. The attention level value directly from the headset is utilized as the main parameter. With most of the sub-blocks connecting together, the control of the LED light is considered stable most of the time. However, we are controlling a robotic car rather than a LED as our final goal. The robotic car requires more than two controls; therefore, attention value is not enough for achieving multiple controls due to the limitation. Alpha and Beta waves are important and have to be involved for replacing attention level.

One of the biggest challenges so far is that the Neurosky headset we are using only contains one sensor. Even though we knew it was a disadvantage before we ordered it, more unsolvable difficulties and challenges popped up as we tried to achieve accurate multiple controls. Throughout the course of the semester, we have been trying to look at multiple controls with Neurosky or one sensor. However, a BCI system with one sensor is usually an SSVEP-based system, in which the sensor will be placed on the occipital lobe of the brain to discover such an ERP signal [16]. In addition, the sample rate of the headset is fixed to be 512Hz, and the calculated values are outputted per second, which is considered as slow if we want to minimize the time-delay.

Thus, after the MDR, we discussed this concern with our faculty advisor, Professor Xia, and our group has voted and reached a conclusion: if we want to control the robotic car to do more than stably moving forward or stop using EEG, it is necessary for us to have a headset with more sensors. Emotiv is the only commercial headset in the market that has multiple sensors covering at different regions of the brain [7]. Although we will need to interface the headset with our software and research on the stimuli that can be use in this headset, it is more promising to yield a reliable multiple command classification, as we can reference some of the BCI system to look at.

Acknowledgment

We would like to thank Prof. Xia for providing feedback to our design, for sponsoring us to purchase the Emotiv headset, and being our advisor. We also want to thank Prof. Mettu and Mr. Alexander de Geofroy for providing us an overview of the Project Brainmobile, as well as background knowledge and the suggestion on EEG. We also want to thank Fran for ordering the part for us, and M5 for providing Arduino and XBEE from the past SDP project.

References

[1] Disability Funders Network. “Disability Stats and Facts.” 2012. Web Accessed November 10, 2014.

[2] John Hopkins Medicine. “Statistics of Disability.” 2013. Web Accessed November 10, 2014.

[3] Brain–computer interfaces for communication and control Jonathan R. Wolpawa,b,*, Niels Birbaumerc,d, Dennis J.

[4] "MindWave Mobile." NeuroSky Store —. N.p., n.d. Web. 15 Dec. 2014. <http://store.neurosky.com/products/mindwave-mobile>.

[5] F Lotte, M Congedo,AĹecuyer, FLamarche and B Arnaldi. A review of classification algorithms for EEG-based brain–computer interfaces. N.p.,n.d. Web. Dec. 2014. <http://iopscience.iop.org/1741-2552/4/2/R01/pdf/1741-2552_4_2_R01.pdf>.

[6] Hassib, Mariam. Mental Task Classification Using Single-electrode Brain Computer Interfaces. Thesis. Universität Stuttgart, 2012. N.p.: n.p., n.d. Web. <http://elib.uni-stuttgart.de/opus/volltexte/2012/7982>.

[7] "Emoiv" Emotiv. N.p., n.d. Web. 15 Dec. 2014. <https://emotiv.com/>.

[8] "Bluetooth High Speed Technology." High Speed | Bluetooth Technology Website. N.p., n.d. Web. 15 Dec. 2014. <http://www.bluetooth.com/Pages/High-Speed.aspx>.

[9] "Introduction." Thinkgear.net_sdk_dev_guide_and_api_reference [NeuroSky Developer. N.p., n.d. Web. 15 Dec. 2014. <http://developer.neurosky.com/docs/doku.php?id=thinkgear.net_sdk_dev_guide_and_api_reference>.

[10] "Brainwave Starter Kit Product Support." Mindwavemobile [NeuroSky Developer. N.p., n.d. Web. 15 Dec. 2014. <http://developer.neurosky.com/docs/doku.php?id=mindwavemobile>.

[11] Larsen, Erik A. Classification of EEG Signals in a Brain- Computer Interface System . N.p.,n.d. Web. Dec. 2014. <http://www.diva-portal.org/smash/record.jsf?pid=diva2:440513>.

[12] Wu, Jia (School of Computer Science, China University of Geosciences, Wuhan, China); Pan, Shirui; Zhu, Xingquan; Cai, Zhihua; Zhang, Peng; Zhang, Chengqi Source: Expert Systems with Applications, v 42, n 3, p 1487-1502, February 15, 2015 http://www.engineeringvillage.com.silk.library.umass.edu/search/doc/abstract.url?&pageType=quickSearch&searchtype=Quick&SEARCHID=19f9187cM1abaM45a3Ma021M436697e048d4&DOCINDEX=1&database=1&format=quickSearchAbstractFormat&tagscope=&displayPagination=yes

[13] "XBee® ZB." - Digi International. N.p., n.d. Web. 15 Dec. 2014. <http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/zigbee-mesh-module/xbee-zb-module#specs>.

[14] "Statistical Analysis Software, SAS/STAT." *Statistical Analysis Software, SAS/STAT*. N.p., n.d. Web. 15 Dec. 2014. <http://www.sas.com/en_us/software/analytics/stat.html>.

[15] Yue Liu; Xiao Jiang; Teng Cao; Feng Wan; Peng Un Mak; Pui-In Mak; Mang I Vai, "Implementation of SSVEP based BCI with Emotiv EPOC," Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS), 2012 IEEE International Conference on , vol., no., pp.34,37, 2-4 July 2012
doi: 10.1109/VECIMS.2012.6273184
URL:
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6273184&isnumber=6273175

[16] Luo, A, and T.J Sullivan. "A User-Friendly Ssvep-Based Brain-Computer Interface Using a Time-Domain Classifier." *Journal of Neural Engineering*. 7.2 (2010). Print.