CONRAD SOIC 2015 (HEALTH AND NUTRITION) - TEAM MEDEX

PORTFOLIO SUMMARY

Team Members:
Rohan Deshpande rohanvdeshpande@hotmail.com
WenTao Zhang wentaozhang15@gmail.com

Team Mentor:
Vinayak Deshpande deshpa52@yahoo.com

Team Coach:
Anagha Kalvade anagha29@yahoo.com

School:
East Chapel Hill High School

City: State:
Chapel Hill North Carolina

Immediate Medical Response System (iMrs)

Our technology, the Immediate Medical Response System (iMrs) is an inexpensive, comprehensive, and versatile medical emergency-response system that appeals to both developed and developing nations. iMrs is a 3-part system: an Mtach biosensor that can detect a medical emergency, an Mdoc database software that analyzes patients' vital information and diagnoses medical emergencies, and an Mrotor mini-Unmanned Aviation Vehicle (UAV) that responds to the emergency alerts and acts as a first responder system that delivers elementary, life-saving treatment to the patient. iMrs will cut the time it takes for medical attention to reach the patient, a critical issue in developing nations, and save many lives. iMrs is also cheap, comprehensive, and versatile, making it attractive in developed countries to deliver fast emergency response at low cost.

PORTFOLIO ELEMENT A

PRESENTATION AND JUSTIFICATION OF THE PROBLEM

According to the World Health Organization, the top two causes of death in the world are ischemic heart disease and stroke. Together, these two medical emergencies result in over 14 million deaths annually worldwide. By 2015, 3 million more people are projected to die from these diseases. Both medical emergencies require immediate medical attention to maximize the patient's chance for survival. In America alone, 130,000 of the 800,000 patients who have stroke die during the first ten minutes after a stroke. Land and aerial helicopter ambulances are currently the most common methods to respond to medical emergencies. Land ambulances, however, are susceptible to circuitous routes, traffic jams, and limited access to certain areas such as trails. The EMS response is worst in developing countries, where average response times range anywhere from one hour in Pakistan to a little under three hours in West Azerbaijan. Meanwhile, aerial ambulances can fly directly to the patient in a straight-line path. However, the helicopter ride's astronomical price, which can cost patients and insurance companies well over $45,000, combined with its need for a flat landing area, makes it an even less feasible option than traditional ambulances.

This current plight calls for an inexpensive, comprehensive, and versatile medical emergency-response system that appeals to both developed and developing nations. The iMrs is designed specifically to fill that niche. Our technology, the Immediate Medical Response System, iMrs, is a 3-part system: an Mtach biosensor that can detect a medical emergency, an Mdoc software that analyzes patients' vital information and diagnoses medical emergencies, and an Mrotor mini-Unmanned Aviation Vehicle (UAV) that responds to the emergency alerts and acts as a first responder system that delivers elementary, life-saving treatment to the patient. iMrs will cut the time it takes for medical attention to reach the patient, a critical issue in developing nations and save many lives. iMrs is also cheap, comprehensive, and versatile, making it attractive in developed countries to deliver fast emergency response at low cost.

PORTFOLIO ELEMENT B

DOCUMENTATION AND ANALYSIS OF PRIOR SOLUTION ATTEMPTS

Team MedEx's Immediate Medical Response System (iMrs) technology and products belong to the category of medical response products, and more specifically to the subcategory of emergency rapid medical response products. However, given the unique features of the iMrs not offered by existing products, such a categorization is unfitting. Because iMrs is not a single product but rather a system comprising of three main components, we have conducted thorough literature searches for each component to ensure the novelty of our product's design.

Mtach

We looked into eight biosensors that had been patented or are already on the market. Below, we evaluate the three sensors that most closely resemble our product.

Body Patch for Non-invasive Physiological Data Readings
(US Patent Office, Publication Number: WO2008097652 A2)

Description: This device comprises of eight different biosensors located on different parts of the body. A sensor reading is taken every minute to track the patient's vital signature and the data is recorded in a flash drive carried by the patient. The data can be downloaded from the drive for third-party review.

Strengths: The body patch is non-invasive, which lowers the cost and risk associated with using the biosensor. The array of biosensors located around the body will provide doctors with a comprehensive record of the patient's vital information.

Shortcomings: The product is comprised of 8 different sensors located on different parts of the body, which makes the body patches inconvenient to wear. More importantly, the data cannot be wirelessly transmitted to an online database, thereby making emergency-situation detection impossible. Finally, the slow sensor-refreshing speed is unsuitable for emergency detection since physical damage may occur within minutes of an emergency situation.

Disposable Biometric Path Device
(US Patent Office, Publication Number: US20140275932 A1)

Description: The HealthPatch created by Vital Connect is a miniature health monitoring system that uses electronic modules coupled with patch devices to collect vital information from a user. The patch device is disposable and the electronic module (i.e. the biosensor) is reusable for up to its battery life of 72 hours.

Strength: The HealthPatch is the smallest and most versatile biosensor currently on the market. Vital Connect claims that the HealthPatch is about the same weight and size as a Band-Aid and that the biosensor can wirelessly transmit the patient's vital information to a smartphone or computer.

Shortcomings: The short battery life, coupled with the high cost of the product, would make this product unsuitable for marketing in locations where people cannot afford disposable biosensors. Even in its prototype phase, each biosensor costs over $300, which is too much for most people in developed and developing nations to afford.

Mesh Network Stroke Monitoring Appliance
(US Patent Office, Public Number: US20090318779 A1)

Description: Wireless monitoring system for patient monitoring. The monitoring device has a wearable appliance that allows it to detect heart attacks or strokes. After the device detects cardiac failure, it sends distress signals to nearby hospitals via a wireless network.

Strength: The entire sensor is condensed to a wristwatch-like appliance that allows the wearer to move around without significant limitations. The sensor alerts hospitals or first-aid personnel as soon as it detects an emergency.

Shortcomings: The device is very limited in the range of emergencies that it can diagnose since it only detects heart attacks and strokes. Thus, if the patient had an allergic reaction or had fallen to the ground, the sensors would not have recognized the emergency. More importantly, the device does not keep track of a patient's vital information but rather only detects emergencies as they happen. Thus, doctors would not be able to see the patient's vital signature leading up to the emergency.

Mdoc

Due to the novelty of the concept of an automated medical diagnostics system that can analyze a patient's vital information and accurately diagnose medical emergencies, we were only able to find two patents that resembled our product.

Automated Personal Medical Diagnostic System, Method, and Arrangement
(US Patent Office, Publication Number: WO2013066642 A1)

Description: Automated personal medical diagnostic system and arrangement, including a computing device configured to process a portion of the patient's vital information and generation an elementary diagnostic report based on the known information.

Strength: The system can automatically generate simple reports based on known vital signature patterns.

Shortcomings: The accuracy of the reports is low and is not reliable enough for medical professionals not to perform manual diagnostics to guarantee its accuracy. In addition, the system can only diagnose simple medical emergencies (e.g. oxygen deficiency/choking, fever) and not more complicated emergencies (e.g. strokes, cardiac arrest).

Mrotor

Cooperative Grasping and Transport Using Multiple Quadrotors
Distributed Autonomous Robotic Systems, 2013, Volume 83
ISBN : 978-3-642-32722-3
Daniel Mellinger, Michael Shomin, Nathan Michael, Vijay Kumar

Description: Controlling multiple quadrotors to cooperate in order to grasp and transport large payloads. Controlling of the quadrotors are decentralized. In an array of quadrotors, each quadrotor is controlled by measuring the distance between each other, yet the system is able to maintain anonymity between quadrotors. Since the quadrotors are agnostic to their neighbors and have a decentralized control, large arrays of quadrotors can be controlled at the rapid speed of 600 times a second. This gives enough processing for quadrotors to break formations due to obstacle and reform them.

Strength: Allows quadrotors to carry large weights cooperatively and creates more sophisticated and effective communication between multiple quadrotors that increase the productivity of these quadrotors.

Shortcomings: Multiple quadrotors are needed to carry heavier weights instead of having on single quadrotor carry a full load. Having multiple quadrotors carry a single load isn't economical since each quadrotor has a standard high cost. If one quadrotor were to carry it, while the quadrotor might slightly loose some agility that isn't needed for the mrotor, it would simply need slightly more expensive rotors for more thrust.

Autonomous Multi-floor Indoor Navigation with a Computationally Constrained MAV

Robotics and Automation (ICRA), 2011 IEEE International Conference on
ISBN 978-1-61284-386-5
Michael, Nathan ; Kumar, V.

Description:
Autonomous navigation with UAVs in indoor environments allowing quadrotors to map out a deviation from a predefined path due to unexpected obstacles in the GPS database. This article specifically addresses multi-floor mapping with autonomous control and planning for areas that may contain low vertical clearance or strong external forces like air vents.

Strength: Allows quadrotor to pick an appropriate spot to land near the bystander and patient but without crashing into unexpected obstacle such as other people. The quadrotor is able to adjust for other obstacles could be faced such as having to go around a telephone or street lamp when descending in preparation for landing.

Shortcomings: This technology uses large amounts of processing. When the quadrotor is high in the air, facing obstacles is less likely, therefore quadrotor should be able to decide when this Mapping is needed and when it is not.

PORTFOLIO ELEMENT C

PRESENTATION AND JUSTIFICATION OF SOLUTION DESIGN REQUIREMENTS

The Immediate Medical Response System (iMrs) consists of three parts: an Mtach biosensor that records and uploads a patient's vital information to a secure online database, an Mdoc that detects irregularities in a patient's vitals and diagnoses the medical emergencies, and an Mrotor quadcoptor that delivers elementary, life-saving medical equipment (e.g. defibrillators, chest compression systems) to the patient's location. As soon as the Mdoc detects aberrant health behavior from its user, it will automatically relay the user's location provided by the Mtach to an Mrotor dispatcher. The Mrotor will then carry the appropriate medical equipment to the scene of the emergency to aid the patient.

A) Mtach

The Mtach is a noninvasive biosensor that automatically collects vital information from its users and transmits the data to a secure online database easily accessible only by doctors. The Mtach must automatically and continuously transmit the data in order to save time between detecting the emergency and dispatching the Mrotor to the patient's location.

Prioritized list of constraints for Mtach:

1) **Inexpensive** The Mtach should be affordable to users in both developed and developing countries. By evaluating the GNI (Gross National Income) per capita index of these countries, we determined that the average income of a person in a developed country is $25,000 per year and that of a person in a developing country is $3,000. We set the target price of our product at around $200 so that more people can benefit from our technology. While our product may cost more than $200 in its initial stages of development, we believe that we can lower the price of production if we mass produce our product and receive subsidies from government or NGOs (humanitarian organizations, private donors).
2) **Non-invasive** The Mtach will not be surgically implanted into the patient's body. A special adhesive will bind the sensor to the patient's chest to eliminate the risks and costs associated with surgery. In addition, an external sensor is much more easily replaceable than one that is implanted inside the body.
3) **Lightweight** The Mtach should weigh less than half a pound so as not to restrict the wearer's freedom of movement. A large portion of the product's weight will come from the battery. We hope to work in conjunction with the researchers at the NC State Wearable Technology Laboratory to make the Mtach even lighter in the future.
4) **Long-lasting battery** Currently, the batteries used to power the Mtach can last only for 72 to 96 hours on a single charge. We hope that in the future, the Mtach can be powered by a thermoelectric generator that would both eliminate the need to charge the sensor with an outside power source and decrease the weight of the products.

B) Mdoc

The Mdoc receives the information sent to it by the Mtach and records it in a secure online database that only doctors can access. This allows doctors to review the medical history of the patient and diagnose the illness based on the patient's vital information. By comparison, the Mdoc will serve as the patient's blackbox so that doctors can review all pertinent data relating to the medical emergency.

Prioritized list of constraints for Mdoc:

1) **Automatic diagnoses** Mdoc will be able to automatically generate diagnoses based on an incoming patient's vital information by comparing the vital signatures of that patient to those of patients who have suffered similar medical emergencies.
2) **Dispatch** As soon as the Mdoc detects a medical emergency, it will transmit the patient's current location to the drone closest to the patient as well as to the nearest ambulance so that the patient can receive aid as soon as possible.
3) **Secure and accessible database** Data network must be secure and easily accessible to doctors, allowing them to reference data transmitted by the Mtach. This gives doctors accurate quantitative data to construct a timeline of events and other parameters using the patient's vital information for the period of time leading up to the patient's medical emergency.

C) Mrotor

The Mrotor is the capstone of the Immediate Medical Response System. After the Mdoc diagnosis a patient with a medical emergency, it will command the Mrotor to autonomously deploy and travel to the patient's location with the proper medical equipment to negotiate the medical emergency. For example, if the patient went into cardiac arrest, then the Mrotor will fly to the patient (using the Mtach's GPS location feature) with a defibrillator. Likewise, if the patient had a severe allergic reaction, then the Mrotor would carry an EPI-pen to rescue the patient.

Prioritized list of constraints for Mrotor:

1) **Carry a medium-sized payload** The Mrotor should be able to carry a 1.5 kg package since most first-responder emergency medical equipment weighs around 1.0 kilogram. Special equipment, such as defibrillators and chest compression systems, will contribute additional weight to the

package. In the future, we hope to increase the carrying capacity of the quadrotor to 3.0 to 5.0 kilograms.

2) **Fast flying speed** In order to prevent the patient from undergoing brain damage, the Mrotor must reach the patient within 6 minutes of the cardiac arrest. Currently, the fastest drone on the market can fly at 70 miles per hour. The Mrotor should have a sustainable flight speed of 50 miles per hour in most weather conditions so that it may reach the patient before the patient suffers from brain damage.

3) **Long distance flight** The Mrotor should be able to complete a trip of 5 miles on a single charge while carrying the maximum payload. In our initial stages, we hope to place the Mrotors in hospitals that are near or in heavily populated cities. In the future, we may be able to set up Mrotor outposts in less densely populated areas so that the Mrotors can reach out to a larger segment of the population.

4) **Autonomous flight** The Mrotor should be capable of autonomous flight. While the majority of quadrotors on the market today require human piloting, the Mrotor should be able to travel to the patient's location using GPS coordinates transmitted by the patient's Mtach. After the Mrotor finishes its missions, it will return to its home base autonomously.

5) **Communication device** An on-board communication device, such as a camera or speaker, should be present to facilitate doctor-patient communication. If the patient is unconscious or does not know how to operate the emergency medical device on-board the Mrotor, the doctor could give operating instructions to a bystander.

PORTFOLIO ELEMENT D

DESIGN CONCEPT GENERATION, ANALYSIS, AND SELECTION

The decision matrices located below show the side-by-side comparisons that we made for the Mrotor and the Mtach.

Mrotor

When we were designing the Mrotor, we wanted to pick the current leading quadrotor design on the market and modify the quadrotor to accommodate our needs until we receive funding to design and build our own unique Mrotor. After establishing a few basic criteria, we narrowed our selection choices to four commercial quadrotor designs: Phantom 1, Phantom 2, Parrot Drone 2, and the OFM Seeker. We started with a list of the 30 most popular commercial quadrotors and narrowed down to the top 4 based on max speed. We evaluated each of the designs using price, maximum payload, and certain preprogrammed functions, among others features to determine which quadrotor would best serve the role of an Mrotor. In the end, our analyses showed that the OFM Seeker was our best candidate.

	Phantom 1	Phantom 2	Parrot Drone 2	OFM Seeker
Price	$500	$1,200	$500	$1,500
Max. Speed	16 m/s	16 m/s	11 m/s	32 m/s
Max. Payload Weight	1.0 kg.	1.3 kg.	0.7 kg.	1.5 kg.
Flight Time	12 min.	15 min.	15 min.	20 min.
5 Mile Time	8.0 min.	8.0 min	12 min.	4.0 min.
Low Battery Failsafe	✖	✔	✖	✔
GPS Module	✔	✔	✔	✔
On-board Camera	✖	✖	✔	✔
"Go-Home" Command	✔	✔	✔	✔
Auto-stabilization	✔	✔	✔	✔
Altitude Lock	✔	✔	✔	✔
Course Lock	✔	✔	✔	✔

Mrotor Renders

Top

Isometric

Isometric

Bottom

For a demonstration video of how the iMrs functions, click below.
https://www.youtube.com/watch?v=3-7uXUEbLLA

Mtach

We compared our Mtach design to two leading biosensors on the market today. Using the prioritized criteria that we established in Section C of our Engineering portfolio, we can see that our Mtach design could potentially outcompete VitalConnect's HealthPatch and the Mesh Network Stroke Monitoring Appliance (please see Section B for detailed literature review of these products).

	HealthPatch (VitalConnect)	Mesh Network Stroke Monitoring Appliance	Mtach
Inexpensive	>$300	>$500	<$300
Non-invasive	✔	✔	✔
Lightweight	✔	✔	✔
Long-lasting Battery	✖	✖	✔
Wide Variety of Sensors	✔	✖	✔

Mtach Renders

Top

Isome
tric

PORTFOLIO ELEMENT E

APPLICATION OF STEM PRINCIPLES AND PRACTICES

While we were brainstorming design criteria for Mtach, we established communication with the NC State ASSIST Wearable Technology Laboratory and have received feedback on our Mtach designs. ASSIST is currently developing a self-powered wristband and chest patch, with a mouth-based platform soon to follow. These testbeds contain the ultra low-power SoC developed in Thrust 4, and each has a unique set of sensors because of its location and accessibility to various health and environmental parameters. These works done at ASSIST demonstrate that the technologies needed to develop the Mtach are feasible. We have made improvements to our Mtach to make it more user-friendly and less burdensome to the wearer. We hope to work in their laboratory to develop prototypes of the Mtach for testing and trials.

The work done by Dr. Vijay Kumar of University of Pennsylvania, School of Engineering and Applied Sciences, who is the leading professor in quadrotor research, has demonstrated that the scientific prinicples behind the quadrotor allow it to be implemented feasibly. Both the coach and a member of Team MedEx have connections to the university and hope to establish contact with Dr. Kumar in the near future to in order to receive feedback on how we might be able to modify or even build a quadrotor to best fit the roles of Mrotor.

Throughout the engineering design process, we worked closely with our school's PLTW engineering teachers, Mr. Patrick Pudlo and Mr. William Vincent, to ensure that our designs followed engineering principles and that our design was as cost and material efficient as it could reasonably be.

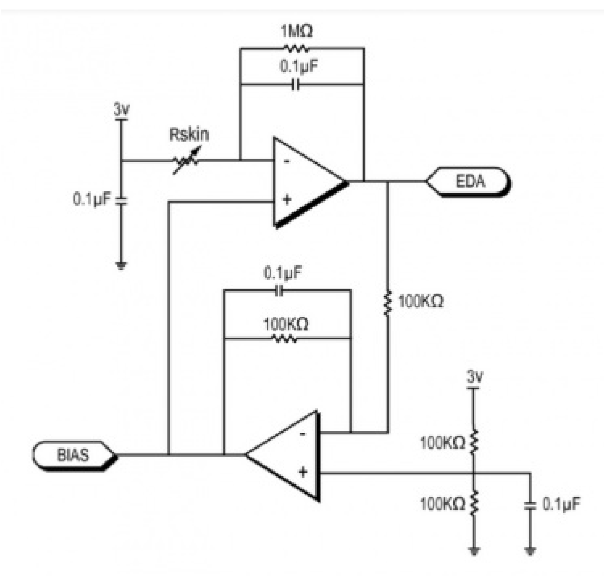


Fig. 1.1

Mtach

The Mtach will detect a patient's heart rate, heart beat variability, motion, and body temperature via EDA (electrodermal activity). Because certain medical emergencies, such as cardiac arrest and stroke, often cause significant changes in skin conductivity due to rapid perspiration, the EDA could easily detect the fluctuation of electrodermal activity and analyze the medical emergency based on the patient's vital signature. In the schematic below, the two power sources send miniscule charges through the patient's skin and detect the resistance of the skin. As moisture levels on the skin increases, resistance decreases. If a patient entered cardiac arrest, the Mtach would sense a rapid decrease in resistance on patient's dermis and would alert the Mrotor and nearby hospitals if more vital signatures confirmed that the patient had indeed entered cardiac arrest.

Mrotor

The primary components of the Mrotor quadrotor are fixed-pitch propellers (angle of propeller cannot be changed), electric motors, and the outside frame. The propellers and motors will be placed equidistant from the center of the quadrotor to ensure balance during flight. The primary choice of material for the airframe will be carbon fiber composites because it offers both structural integrity and light-weight.

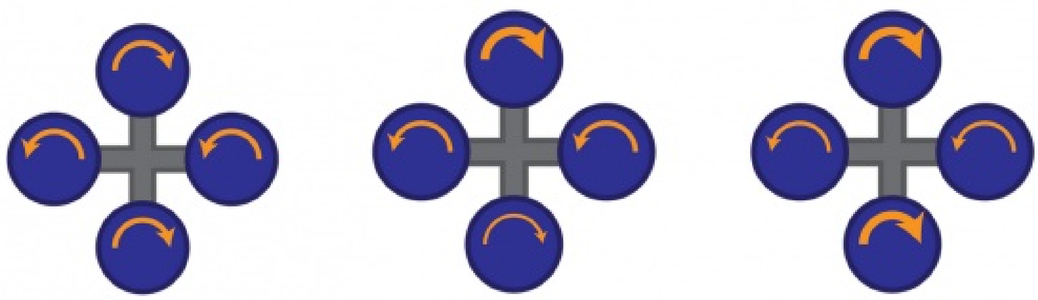
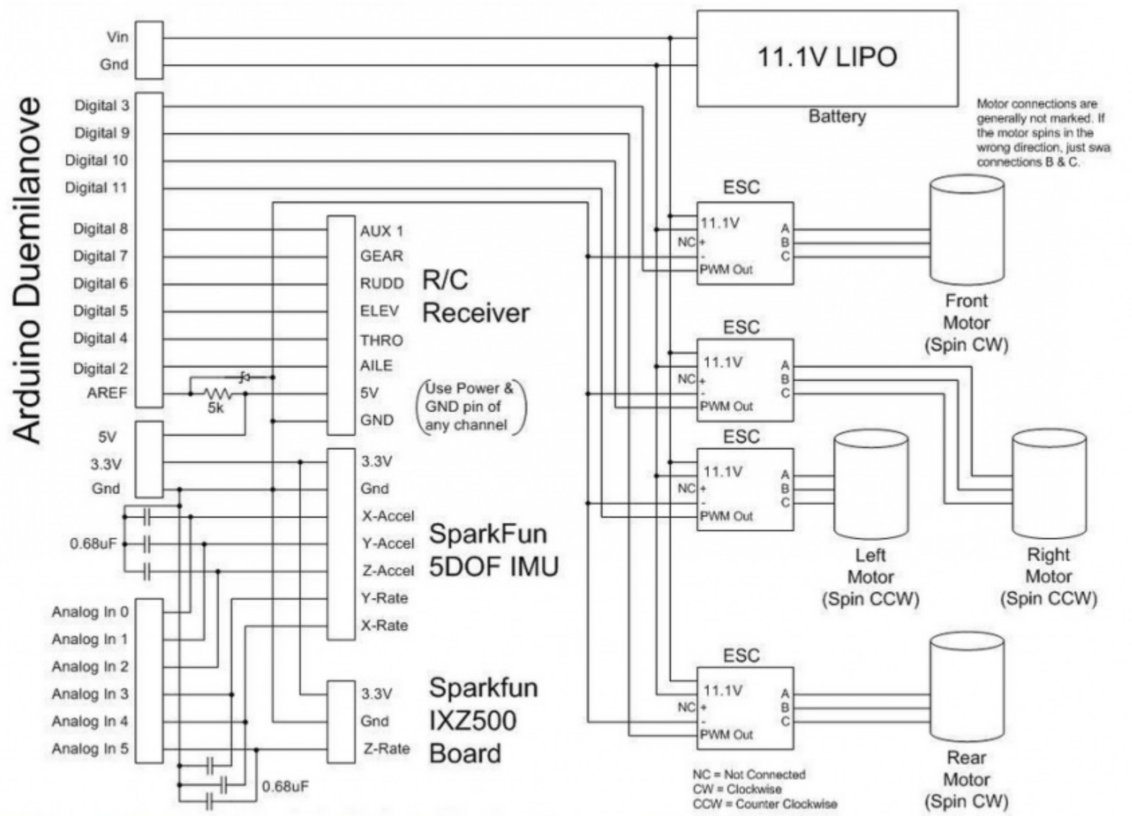


Fig. 2.1 Fig. 2.2 Fig. 2.3

Source: http://cog.yonsei.ac.kr/quad/quad.htm

Quadrotors control thrust, pitch, and yaw (three dimensions of flying objects) by changing the speeds on its rotors. The first image (Fig. 2.1) illustrates a quadrotor changing its pitch by increasing the speed of one motor while keeping the speed of the other three constant. The second image (Fig. 2.2) shows how to move a quadrotor in a vertical direction. By increasing or decreasing the speeds of all the rotors, the quadrotor is able to increase thrust on its vertical axis. If the quadrotor is tilted relative to the ground prior to the increase of thrust on all rotors, the quadrotor will move vertically on this tilted axis. Change in yaw is seen in the third image (Fig. 2.3). By increasing the speed of two opposite rotors, the quadrotor is able to change it's yaw in the direction that the two faster rotors are turning. Notice that any two adjacent rotors are always turning in opposite directions.



As seen schematic above the dissection of the quadrotor is quite simple. On the left side we have an Arduino Uno Rev3.0, a microcontroller made for easy prototyping with 13 I/O pins. In the middle column we have three parts: R/C receiver, SparkFun 5DOF IMU and Sparkfun IXZ500 board. The 5DOF IMU contains a three-axis gyroscope and a digital accelerometer allowing the quadrotor to calibrate and fly with stability. The IXZ 500 is designs specifically to sense pitch and yaw. In the right most column we have four electronic speed controllers to control the speed for the four motors and a 11.1V Lithium battery to supply enough voltage to the motors. Most motors have minor differences in manufacturing. These differences cause changes in motor speeds resulting in thrust, pitch, and yaw. This demands an ESC (Electronic Speed Control) system, which can vary an electric motor’s speed and change the direction and possible serve as a dynamic brake on the quadrotor. Alternatively, our quadrotor could use a gyroscope and an accelerometer to calibrate itself before taking flight.

PORTFOLIO ELEMENT F

CONSIDERATION OF DESIGN VIABILITY

Mtach

Prioritized list of obstacles for Mtach:

1) **Thermoelectric generator** In our current design, the Mtach is powered by a traditional battery since using a battery was the most cost-effective solution. However, the primary drawback to this design is that the battery would have to be replaced every 72 to 96 hours. In the future, we hope to replace the battery power source with a miniature thermoelectric generator. The thermoelectric generator, which is currently in development at the NC State ASSIST Wearable Technology Laboratory, will use the heat of the human body to generate a sufficiently large current to power the Mtach so that the user will not longer have to replace the battery. We are in the process of contacting researchers at the lab and hope to integrate their technology into our sensor in the near future.
2) **Smaller sensor** Currently, the Mtach accommodates only four sensors that include blood pulse sensor, temperature sensor, fall detection sensor, and electrodermal activity (EDA) sensor. The reason the Mtach can hold no more than the four sensors at this development stage is that the sensors are all manufactured separately and must be connected externally to the transmitter. In the future, with more development and prototyping, we hope to shrink the size of the Mtach by directly building and integrating the sensors onto the transmitter, thereby removing the external features of the sensor and allowing us more room to fit other sensors such as allergy and blood sugar detection systems.
3) **Data transmissions** A large component of our design is based on automatic and secure transmission of data from the Mtach to the online database for the Mdoc to analyze. In order for the user's vital information to be uploaded to the database, Wifi or 3G/4G networks should be present in the user's surroundings. While this may not present a problem to people living in developed countries with Wifi, this is a large obstacle for those who live in developing countries where wireless internet connection is not readily available. We hope to address this issue by allowing users to turn their cellphones into hotspots for data transmission so that the patients' vital information can be sent via cellphone signal to the secure database.

Mdoc

Prioritized list of obstacles for Mdoc:

1) **Variable data collection** Currently, the Mtach is programmed to collect and upload a patient's vital signature every minute regardless of whether the Mdoc detects an emergency. In the future, we hope to program Mdoc such that it could automatically command the Mtach to increase broadcasting rate when it detects a medical emergency to ensure that a greater volume of data is recorded for future analysis. Such a function can

be added by using a simple delay algorithm. When the Mdoc tells the Mtach that there's a medical emergency, the value inside the delay function drops. This causes the loop in the program to cycle at a faster rate and allows Mtach to broadcast data at a faster rate.

2) **Establish secure database** The Mdoc must be a secure database accessible by medical professionals/doctors and not third party reviewers (e.g. insurance companies). By having the Mdoc accessible to the doctors, the doctors are able to have a full timeline with quantitative evidence. Instead of asking bystanders or the patient what happened, the doctors have concrete data to base their diagnostics off of. The doctors will also be able to reference past medical emergencies from a patient to look for repetition that can help with diagnosis.

Mrotor

1) **FAA regulations** In order to avoid breaking FAA regulations, Mrotor must carry a database of its region's regulated air spaces. One of FAA primary regulations is that a flying object cannot enter regulated airspace or fly above a ceiling of 400 feet without clearance from traffic controllers and special permits. Therefore, it is imperative that the drone knows where to and where not to fly. A simple GPS system designed for aircrafts would be a simple solution to this problem. The system would be able to direct the quadrotor from flying into restricted airspace.

2) **Autonomously flying** The majority of quadrotors on the market today require human piloting. The Mrotor must be able to fly autonomously to and from the patient's location. In order for the quadrotor to fly autonomously, we need to use the Arducopter mission planner software to allow user to drop waypoint into a GPS map and then converted it into speed signals for each of the 4 rotors of the quadrotor. Doing so will give guide the quadrotor on a straight line path from its dispatch location to the patient's location.

3) **Weather Conditions** Mrotor must be able to fly in all weather conditions to ensure that it can negotiate medical emergencies in all environments. The OFM Seeker is able to fly against tested wind conditions at 20 mph. Further stabilization in rough weather conditions requires use of an auto-stabilization system that can easily be implemented for a small cost.

PORTFOLIO ELEMENT G

CONSTRUCTION OF A TESTABLE PROTOTYPE

Imagine this scenario: A 75-year old senior who lives with his 70-year old wife and 40-year old daughter is walking down the stairs one evening when he feels a convulsion in his chest. He grasps onto the handle and tries to slowly sit down, but the pain was too much to bear. Suddenly, his legs give out and he collapses and tumbles down the stairs. Ten minutes later, his daughter discovers his body at the bottom of the stairs and calls 911, but she is far too late and cannot save his life.

This scenario could have been avoided if the iMrs system had been put to use. Our iMrs system consists of three components: the Mtach, the Mdoc, and the Mrotor. The Mtach noninvasive biosensor, which sends patients' vital information to an online database three times per second, would have transmitted aberrant data when the patient went into the medical emergency. The Mdoc, which diagnoses emergency situations by analyzing patients' vitals, would have immediately detected the cardiac failure and fall and would have relayed the victim's location to the nearest Mrotor and hospital. The Mrotor, which is an autonomous quadrotor that carries an automated external defibrillator, would arrive at the scene of the emergency before the victim suffers brain damage from lack of oxygen and allow bystanders to possibly resuscitate the victim.

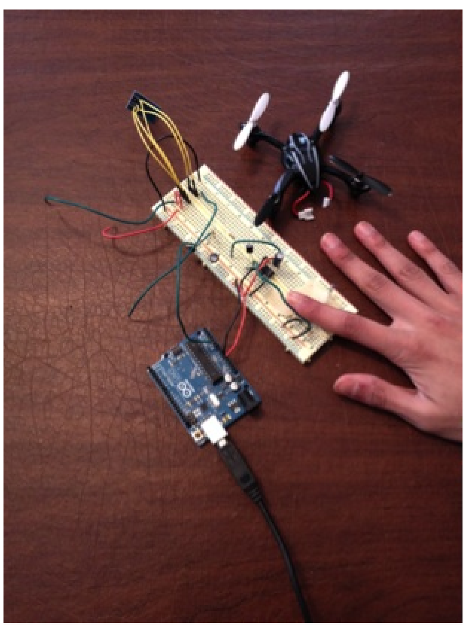


Figure G-1: This image shows the prototype that we have built using the three components of the iMrs system: the Mtach (white breadboard in the center), the Mdoc (Arduino module in the bottom left corner), and the Mrotor (quadrotor in the top right corner). When a person places his or her finger on the Mtach, it will automatically transmit data through the ESP Wifi module (labeled by the red arrow) to the online Mdoc database, where the Mdoc will diagnose the emergency situation based on know patterns and dispatch a quadrotor to deliver an emergency "care package" to the victim. Together, these three inexpensive objects have the potential to save hundreds of thousands of lives each year by providing autonomous and rapid response to medical situations.

Mtach

Vision of future Mtach:

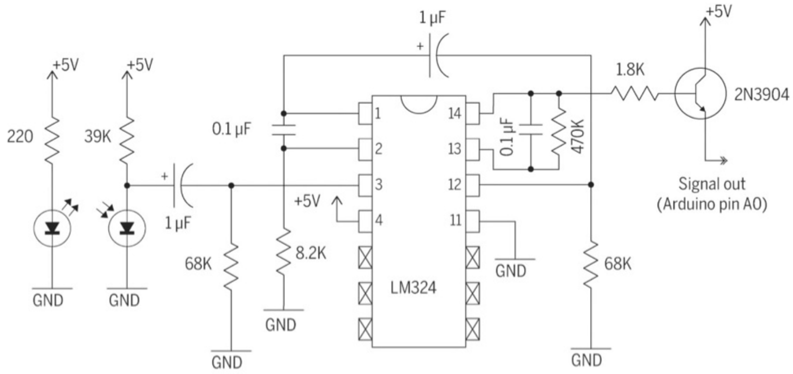


Figure G-2: The circuit diagram depicted above is that of an infrared heartbeat sensor that detects the change in infrared waves given off by blood passing through blood vessels. The data is processed by an IC chip and sent to an Arduino module.

The circuit diagram above shows the functionality of an infrared heartbeat sensor. Once the warm blood leaves the heart to travel to different parts of the body, the blood will give off heat signatures in the form of infrared heat waves. The infrared sensors will emit a pulse of infrared light onto the skin. When the light beam hits the pulses of blood underneath the skin, it is reflected and the infrared sensor detects a voltage change in the circuit. The processing unit/control module (LM324) will pick up the input voltage drop and output a value that corresponds to the smaller voltage. The Arduino module will process the value outputted by the LM324 chip and convert the information to a data point that it will then graph on a heartbeat monitor.

The heart beat sensor is only the tip of the iceberg. By working with the Arduino module, we have discovered its potential to carry multiple other sensors, such as those that measure temperature and fall-detection. In the future, we hope to use the infrared heartbeat detector as a temperature detector as well, since both vital signatures can be easily and accurately measured using infrared technology. Similar to commercial infrared thermometers, our prototype temperature detector will emit and detect an infrared wave and determine the patient's body temperature through the process. We believe that the applications for this technology can be just as broad as the applications for the heartbeat detector – from fever monitoring in infants and children to heat-shock prevention in athletes. While a fall detection monitor (G-force sensor) will be more difficult to incorporate into our design due to the complicated circuit boards, we hope to have such a sensor built into the Mtach along with the heartbeat and temperature sensor because we acknowledge the importance of quickly detecting whether a person has fallen down the stairs or has slipped in the bathtub. If an adult or senior were to fall, then there is a large chance that that person suffered bone damage and possibly even cranial damage. Thus, it is imperative that we detect a fall as soon as it occurs.

Current prototype of Mtach that we have assembled:

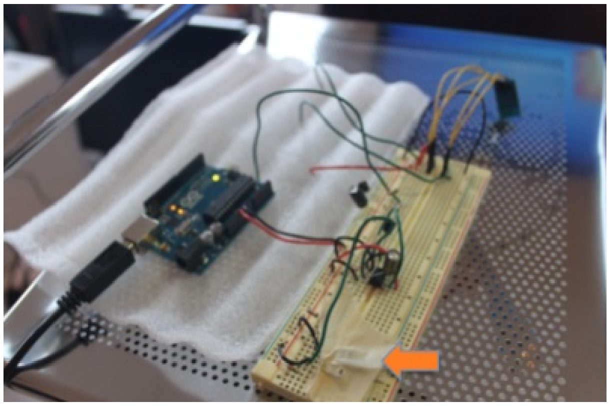


Figure G-3: The orange arrow points to where a person should place his or her finger in order for the Mtach to detect the wearer's heartbeat. In the future, we hope to shrink the sensor into a wearable ring so that it is less of a burden for the wearer.

We have developed a fully functional noninvasive sensor that utilizes an infrared emitter, an infrared detector, and a programmable Arduino module to detect the blood passing through the blood vessels underneath the skin. The minute voltage change is amplified by a capacitor. The current travels through LM-series integrated circuit (IC) chips and then through 2 low pass filters. The increased voltage is then sent to the Arduino module, which collects the fluctuating voltage values created by the heartbeat and constructs a graph that visually depicts the pulse. If the sensor does not detect consistent infrared pulses from the wearer, then it will send out an emergency dispatch signal to the nearest Mrotor and hospital to notify them of the medical emergency.

Mdoc

Vision of future Mdoc:

Our vision of Mdoc is an AI medical emergency detection, diagnosis, and dispatch device that uses algorithms to analyze patients' vital signatures in prior emergency situations and learn from those patterns to analyze and figure out future emergencies. Ideally, we will implement our Mtach and Mdoc system in large hospitals and healthcare facilities across the United States and gather data from patients

Current prototype of the Mdoc we have assembled:

We have programmed the Mtach to automatically update the online medical database with the patient's vital information three times per second. When the Mdoc receives a heartbeat value or temperature reading from the Mtach on the patient, it automatically sends that data to PushingBox (a cloud database that allows users to wirelessly transmit and review data from anywhere in the world) with three commands – one that tells the Arduino module which domain it will be sending the command to, the next that tells the module how many bytes of data it will be sending, and the last that sends the link as an HTTP/1.1 GET request. We are using

PushingBox because we cannot send HTTPS requests with the ESP module. The benefits of using PushingBox over other online databases such as ThingSpeak is that the PushingBox link automatically redirects itself to an HTTPS link of the Google Drive, which then uses an elaborate 60 lines of code to convert this HTTPS link to a command for Google Drive and inserts the data into cells. (Please refer to Appendix G-1 through G-5 for code/further explanations) By coding our Arduino module to automatically send patients' vital information to a password protected Google spreadsheet through the ESP-8266 WiFi module, we allow both patients and doctors to have a real-time record of their patients' vital information.

More importantly, we have developed a functional alert system that immediately sends out an alert to doctors if the Mtach detects an irregularity in the patient's heart beat rhythm. We will conduct more tests to consolidate the reliability of our product and hope to make the Mdoc 95% false-positive accurate in detecting heart attacks or strokes. In the future, we hope to have the Mdoc diagnose more types of medical emergencies, such as high fevers or if a patient falls.

Mrotor

Vision:

If we are fortunate enough to receive the Conrad grant, then we will purchase an OFM Seeker quadrotor and test out its top speed, sustainable flight time, sustainable payload weight, and homing system. Since the quadrotor will be flying in outdoors conditions when it delivers the payload, we will test all specifications outside on the nearby school track. We will have the quadrotor fly a horizontal distance of 100 meters at a constant altitude and time the flight in order to determine its top speed. We will place weights in a box that's similar to the one that carries the automated external defibrillator and fly the quadrotor around at a constant altitude to determine the quadrotor's sustainable flight time and sustainable payload weight. Finally, we will fly the quadrotor from the track to the school's baseball field and command the quadrotor to fly back on its own using its homing system.

Current prototype:

Even though both team members of iMrs have a full understanding of how a quadrotor works and flies, we decided that it was not economical feasible for us to obtain individual parts to build our own quadrotor in this phase of prototyping. Instead, we looked at commercially available models and did a cost-benefit comparison for the leading models currently on the market. After gathering data for the top four commercially available quadrotors, we decided that the OFM Seeker was best suited for the task because of its high maximum speed, high maximum payload weight, long flight time, along with other autonomous homing features that may reduce the amount of coding that we'll have to do in order to get the quadrotor to flight to and from an emergency scene. First and foremost, the OFM Seeker can fly at a sustained top speed of 32 m/s. This means that the quadrotor can complete a 5-mile distance in 4.0 minutes. Because a normal person begins to suffer brain damage 6 minutes after he or she goes into heart failure, having the quadrotor reach the patient in a short period of time is imperative. Next, the Seeker can carry a maximum payload of 1.5 kg when flying at its top speed. Since the only object the quadrotor will be carrying is a 0.75 kg automated external defibrillator (AED), it should be able to complete the flight in four minutes.

References/Citations:

- PushingBox (https://www.pushingbox.com)
- ThingSpeak (https://thingspeak.com)
- Arduino (http://arduino.cc)
- Google Sheets/Script (https://docs.google.com/spreadsheets/u/0/)
- Processing (https://processing.org)

Appendix

Code.gs

```
//-----------------------------------------------
//Originally published by Mogsdad@Stackoverflow
//Modified for jarkomdityaz.appspot.com
//-----------------------------------------------
/*

GET request query:
https://script.google.com/macros/s/<gscript id>/exec?tempData=data_here

*/


/* Using spreadsheet API */

function doGet(e) {
  Logger.log( JSON.stringify(e) );  // view parameters

  var result = 'Ok'; // assume success

  if (e.parameter == undefined) {
    result = 'No Parameters';
  }
  else {
    var id = '1k5fZx9cEIvIHrq4bkID68I94huddnxKxpqZwewuSNMg'   ; // Spreadsheet ID
    var sheet = SpreadsheetApp.openById(id).getActiveSheet();
    var newRow = sheet.getLastRow() + 1;
    var rowData = [];
    //var waktu = new Date();
    rowData[0] = new Date(); // Timestamp in column A
    for (var param in e.parameter) {
      Logger.log('In for loop, param=' +param);
      var value = stripQuotes(e.parameter[param]);
      //Logger.log(param + ':' + e.parameter[param]);
      switch (param) {
        case 'tempData': //Parameter
          rowData[1] = value; //Value in column B
          break;
     // case 'column_C':
       // rowData[2] = value;
       // break;
        default:
          result = "unsupported parameter";
      }
    }
    Logger.log(JSON.stringify(rowData));

    // Write new row below
    var newRange = sheet.getRange(newRow, 1, 1, rowData.length);
    newRange.setValues([rowData]);
  }

  // Return result of operation
  return ContentService.createTextOutput(result);
}

/**
* Remove leading and trailing single or double quotes
*/
function stripQuotes( value ) {
  return value.replace(/^["']|['"]$/g, "");
}
```

Appendix G-1: This is a Google Script code that we wrote that turns an HTTPS link into a command for Google Sheets to collect data and input the data into cells.

AUTOESPT2

```
#include <SoftwareSerial.h>

#define DEBUG true

SoftwareSerial esp8266(2,3); // make RX Arduino line is pin 2, make TX Arduino line is pin 3.
                             // This means that you need to connect the TX line from the esp to the Arduino's pin 2
                             // and the RX line from the esp to the Arduino's pin 3
String Start = "AT+CIPSTART=\"TCP\",\"api.pushingbox.com\",80\r\n";
String val ="20";
String Get1 = "GET /pushingbox?devid=v6A5232E496FBD72&tempData=";
String Get2 ="\r\n";
String GetFinal;
String Send1="AT+CIPSEND=";
String SendLen;
String SendFinal;

void setup()
{
  Serial.begin(9600);
  esp8266.begin(9600); // your esp's baud rate might be different
  sendData("AT+RST\r\n",2000,DEBUG); // reset module
  sendData("AT+CWMODE=1\r\n",1000,DEBUG);
  sendData("AT+CWJAP=\"[redacted]\",\"[redacted]\"\r\n",8000,DEBUG);
                    Username and Password
}

void loop()
{
  sendData(Start,1000,DEBUG);
  GetFinal = Get1;
  GetFinal += val;
  GetFinal += " HTTP/1.1\r\n";
  GetFinal += "Host: api.pushingbox.com\r\n\r\n";
  SendLen = "89";
  SendFinal = Send1;
  SendFinal += SendLen;
  SendFinal += Get2;
  sendData(SendFinal,1000,DEBUG);
  sendData(GetFinal,1000,DEBUG);
  sendData("AT+CIPCLOSE\r\n",1000,DEBUG);
}


String sendData(String command, const int timeout, boolean debug)
{
    String response = "";
}

void loop()
{
  sendData(Start,1000,DEBUG);
  GetFinal = Get1;
  GetFinal += val;
  GetFinal += " HTTP/1.1\r\n";
  GetFinal += "Host: api.pushingbox.com\r\n\r\n";
  SendLen = "89";
  SendFinal = Send1;
  SendFinal += SendLen;
  SendFinal += Get2;
  sendData(SendFinal,1000,DEBUG);
  sendData(GetFinal,1000,DEBUG);
  sendData("AT+CIPCLOSE\r\n",1000,DEBUG);
}


String sendData(String command, const int timeout, boolean debug)
{
    String response = "";

    esp8266.print(command); // send the read character to the esp8266

    long int time = millis();

    while( (time+timeout) > millis())
    {
      while(esp8266.available())
      {

        // The esp has data so display its output to the serial window
        char c = esp8266.read(); // read the next character.
        response+=c;
      }
    }

    if(debug)
    {
      Serial.print(response);
    }

    return response;
}
```

Appendix G-2: This is the Arduino code that we wrote that runs PushingBox. PushingBox converts the HTTP link to HTTPS link so that Google Sheets can store and analyze. It's in this set of commands that you will find the three commands. PushingBox allows users to input data into its database at a rate of 4 data points per second.

AUTOESPF_

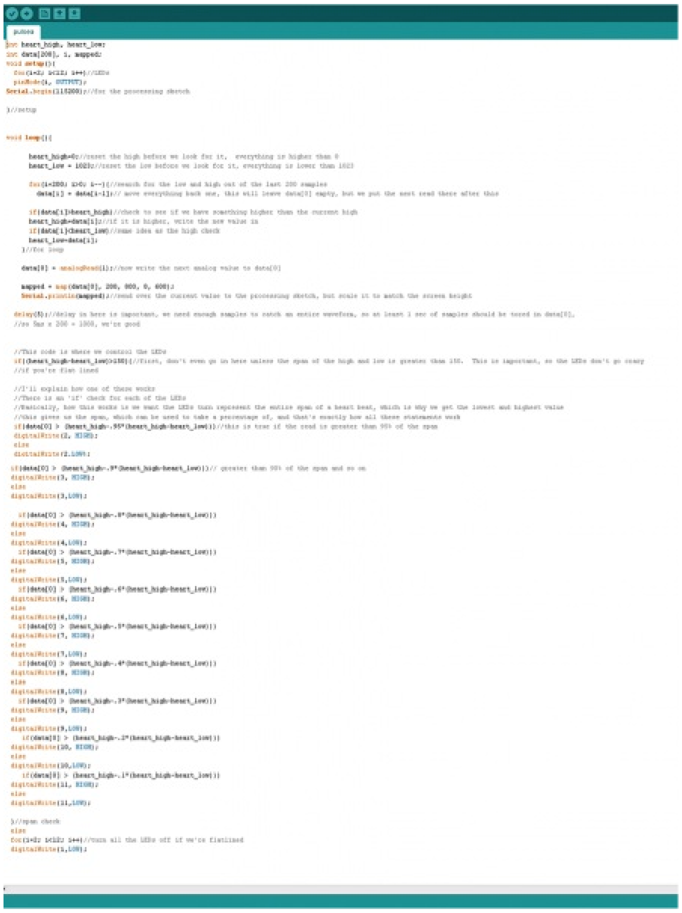
```
#include <SoftwareSerial.h>

#define DEBUG true

SoftwareSerial esp8266(2,3); // make RX Arduino line is pin 2, make TX Arduino line is pin 3.
                             // This means that you need to connect the TX line from the esp to the Arduino's pin 2
                             // and the RX line from the esp to the Arduino's pin 3
String Start = "AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n";
String val ="20";
String Get1 = "GET /update?key=VOSSJQIYL1JQ57NS&field1=";
String Get2 ="\r\n";
String GetFinal;
String Send1="AT+CIPSEND=";
String SendLen;
String SendFinal;

void setup()
{
  Serial.begin(9600);
  esp8266.begin(9600); // your esp's baud rate might be different
  sendData("AT+RST\r\n",2000,DEBUG); // reset module
  sendData("AT+CWMODE=1\r\n",1000,DEBUG);
  sendData("AT+CWJAP=\"[redacted]\",\"[redacted]\"\r\n",8000,DEBUG);
                  Username and Password
}

void loop()
{
  sendData(Start,2000,DEBUG);
  GetFinal = Get1;
  GetFinal += val;
  GetFinal += Get2;
  SendLen = String(GetFinal.length());
  SendFinal = Send1;
  SendFinal += SendLen;
  SendFinal += Get2;
  sendData(SendFinal,2000,DEBUG);
  sendData(GetFinal,2000,DEBUG);
  delay(13000);
}


String sendData(String command, const int timeout, boolean debug)
{
    String response = "";

}

void loop()
{
  sendData(Start,2000,DEBUG);
  GetFinal = Get1;
  GetFinal += val;
  GetFinal += Get2;
  SendLen = String(GetFinal.length());
  SendFinal = Send1;
  SendFinal += SendLen;
  SendFinal += Get2;
  sendData(SendFinal,2000,DEBUG);
  sendData(GetFinal,2000,DEBUG);
  delay(13000);
}


String sendData(String command, const int timeout, boolean debug)
{
    String response = "";

    esp8266.print(command); // send the read character to the esp8266

    long int time = millis();

    while( (time+timeout) > millis())
    {
      while(esp8266.available())
      {

        // The esp has data so display its output to the serial window
        char c = esp8266.read(); // read the next character.
        response+=c;
      }
    }

    if(debug)
    {
      Serial.print(response);
    }

    return response;
}
```

Appendix G-3: This code that we wrote is almost identical to the code that we used for PushingBox. The only difference between PushingBox and ThingSpeak is that maximum refresh rate: ThingSpeak only allow users to update data once every 15 seconds.

pulseae3

```
import processing.serial.*;

Serial myPort;  // Create object from Serial class
int val, screen_increment, old_x=0, old_y=0;      // Data received from the serial port
String inString;  // Input string from serial port
int lf = 10;      // ASCII linefeed
int min=7;
int peak =0;
int mind=0;
int peakd=0;
int thresh= 300;
int pass=1;
int hcolor= 0;
int ecolor=208;
void setup()
{

  size(displayWidth-100, 600);//screen size setup, display width is read into teh program, and I
  //clipped it a little bit.  The screen height is set to be 600, which matches the scaled data,


  myPort = new Serial(this, "COM3", 115200);//Set up the serial port
  myPort.bufferUntil(lf);//read in data until a line feed, so the arduino must do a println
  background(208,24,24);//make the background that cool blood red

}//setup

void draw(
)
{
//nothing in here, this is kind of like the void loop in arduino
}

void serialEvent(Serial myPort) { //this is called whenever data is sent over by the arduino

  inString = myPort.readString();//read in the new data, and store in inString
  inString = trim(inString);//get rid of any crap that isn't numbers, like the line feed
  val = int(inString);//convert the string into a number we can use

    if(val> peak+25)
  {
    if(pass<1)
    {mind=min;
      min=7;
      pass=1;
    }
    peakd=0;
    peak=val;
    min= val;
  }
  if(val< min-25)
  {
    if(pass>0)
    {
      peakd=peak;
      peak=0;
      pass=0;
    }
    mind=0;
    peak=val;
    min=val;
  }
  if(peakd>0)
  {
    hcolor=255;
  }
  if(mind>0)
  {
    hcolor=0;
  }
  if(val>600)
    {
      ecolor=255;
    }
    else{ecolor=24;}
   textSize(32);
  fill( color(208,24,24));
  stroke(color(208,24,24));
  rect(10, 10, 400, 80);
  fill( color(255, 255, 255));
  text(val, 20, 50);
  fill(0);
   fill( color(0, hcolor, 0));
  stroke(color(208,24,24));
  ellipse(140, 40, 50, 50);
  fill( color(208, ecolor, ecolor));
  stroke(color(208,24,24));
  ellipse(220, 40, 50, 50);
  strokeWeight(12);//beef up our white line
  stroke(255, 255, 255);//make the line white

  //here's where we draw the line on the screen
  //we need to draw the line from one point to the next
  //so we have the point we last drew, to the new point
  //values are written as an x,y system, where x is left to right, left most being 0
  //y is up and down, BUT 0 is the upmost point,
  //so we subtract our value from the screen height to invert
   //screen increment, is how we progress teh line through the screen
   line(old_x, old_y, screen_increment, 600-val);

   //store the current x, y as the old x,y, so it is used next time
   old_x = screen_increment;
   old_y = 600-val;

   //increment the x coordinate,  you can play with this value to speed things up
   screen_increment=screen_increment+2;

   //this is needed to reset things when the line crashes into the end of the screen
   if(screen_increment>(displayWidth-100)){
     background(208,24,24); //refresh the screen, erases everything
     screen_increment=-50; //make the increment back to 0,
     //but used 50, so it sweeps better into the screen
     //reset the old x,y values
   old_x = -50;
   old_y = 0;


   }// if screen... blah blah

  '/serial event
```

Appendix G-4: This series of code that we wrote was written using Java in Processing. On the Serial Communications Port, the Arduino module sends the data from the pulse sensor over the Serial Communications Port. The Processing code graphs that data, detects the crest and the troughs of the heartbeat wave, and detect medical emergencies.



```
pulses
int heart_high, heart_low;
int data[200], i, mapped;
void setup(){
  for(i=2; i<12; i++)//LEDs
  pinMode(i, OUTPUT);
Serial.begin(115200);//for the processing sketch

}//setup


void loop(){

      heart_high=0;//reset the high before we look for it.  everything is higher than 0
      heart_low = 1023;//reset the low before we look for it. everything is lower than 1023

      for(i=200; i>0; i--){//search for the low and high out of the last 200 samples
        data[i] = data[i-1];// move everything back one, this will leave data[0] empty, but we put the next read there after this

      if(data[i]>heart_high)//check to see if we have something higher than the current high
      heart_high=data[i];//if it is higher, write the new value in
      if(data[i]<heart_low)//same idea as the high check
      heart_low=data[i];
    }//for loop

    data[0] = analogRead(1);//now write the next analog value to data[0]

    mapped = map(data[0], 200, 800, 0, 600);
    Serial.println(mapped);//send over the current value to the processing sketch, but scale it to match the screen height

  delay(5);//delay in here is important, we need enough samples to catch an entire waveform, so at least 1 sec of samples should be tored in data[0],
  //so 5ms x 200 = 1000, we're good


  //This code is where we control the LEDs
  if((heart_high-heart_low)>150){//First, don't even go in here unless the span of the high and low is greater than 150.  This is important, so the LEDs don't go crazy
  //if you're flat lined

  //I'll explain how one of these works
  //There is an 'if' check for each of the LEDs
  //Basically, how this works is we want the LEDs turn represent the entire span of a heart beat, which is why we get the lowest and highest value
  //this gives us the span, which can be used to take a percentage of, and that's exactly how all these statements work
  if(data[0] > (heart_high-.95*(heart_high-heart_low)))//this is true if the read is greater than 95% of the span
  digitalWrite(2, HIGH);
  else
  digitalWrite(2,LOW);

 if(data[0] > (heart_high-.9*(heart_high-heart_low)))// greater than 90% of the span and so on
 digitalWrite(3, HIGH);
 else
 digitalWrite(3,LOW);

   if(data[0] > (heart_high-.8*(heart_high-heart_low)))
 digitalWrite(4, HIGH);
 else
 digitalWrite(4,LOW);
   if(data[0] > (heart_high-.7*(heart_high-heart_low)))
 digitalWrite(5, HIGH);
 else
 digitalWrite(5,LOW);
   if(data[0] > (heart_high-.6*(heart_high-heart_low)))
 digitalWrite(6, HIGH);
 else
 digitalWrite(6,LOW);
   if(data[0] > (heart_high-.5*(heart_high-heart_low)))
 digitalWrite(7, HIGH);
 else
 digitalWrite(7,LOW);
   if(data[0] > (heart_high-.4*(heart_high-heart_low)))
 digitalWrite(8, HIGH);
 else
 digitalWrite(8,LOW);
   if(data[0] > (heart_high-.3*(heart_high-heart_low)))
 digitalWrite(9, HIGH);
 else
 digitalWrite(9,LOW);
    if(data[0] > (heart_high-.2*(heart_high-heart_low)))
 digitalWrite(10, HIGH);
 else
 digitalWrite(10,LOW);
    if(data[0] > (heart_high-.1*(heart_high-heart_low)))
 digitalWrite(11, HIGH);
 else
 digitalWrite(11,LOW);

 }//span check
 else
 for(i=2; i<12; i++)//turn all the LEDs off if we're flatlined
 digitalWrite(i,LOW);
```

Appendix G-5: Using this code that we wrote, the Arduino module can read the input values and automatically shrink the Processing window to best fit the heartbeat input values. Furthermore, if we were to link ten LED lights in series onto the Mtach, then the lights would go on and off depending on the pulse strength (e.g. more lights for stronger pulses).

PORTFOLIO ELEMENT H

PROTOTYPE TESTING AND DATA COLLECTION PLAN

Mtach

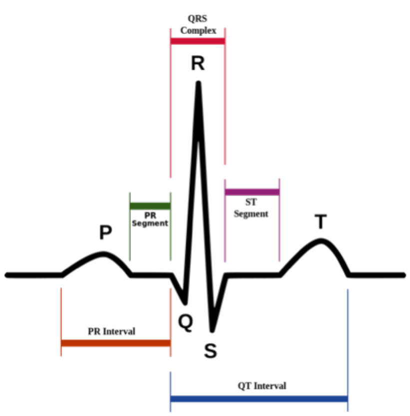
For our current prototype, we set out to establish proof-of-concept for the three most high priority constraints listed in Question C of the portfolio: inexpensive, non-invasive, and lightweight. The prototype that we constructed cost less than $20 to develop (an order of magnitude cheaper than we had anticipated), can detect a patient's vital signature using a noninvasive sensor worn around the patient's finger, and weighs less than 200 grams.

Current prototype:

Pass/Fail Criteria: If the Mtach can accurately detect a patient's heartbeat and send the data to the Mdoc online database at PushingBox, it passes. If not, it fails.

Material:
- Arduino Microcontroller module
- Infrared light emitter and detector
- Capacitors (22 pF, 0.1 mF, 10 mF, 100 mF)
- Resistors(50 Ω, 100 Ω, 10K Ω)
- ESP Wifi module
- LM386 amplifier

Stepwise Procedure:
1. Connect the Mtach to a power source and make sure the Arduino module and Wifi module have a constant power supply.
2. Connect the Mtach to a computer monitor.
3. Place your fingertip over the infrared light emitter and detector and apply constant pressure to the sensor. Give 10 seconds for the sensor to be calibrated properly.
4. Continue to apply constant pressure to the sensor. After the device calibrates, you should see consistent heartbeat waves on the computer monitor just like you would on an EKG. The PR and QT interval will be clearly shown on the screen.
5. A green dot will flash on the screen when the Mtach detects a local maximum (e.g. crest of the wave at R) and a black dot will flash when the Mtach detects a local minimum (e.g. trough of the wave at S). If the Mtach does not detect a heartbeat, then a white dot will flash.



Future Prototype:

For our final prototype, we will add on a temperature sensor, a fall-detection sensor, and a thermoelectric battery that uses the patient's body temperature to power the Mtach device. We will conduct noninvasive tests on a simple random sample of individuals using commercially available sensors and with our own Mtach in order to further determine the accuracy of our pulse, temperature, and fall-detection sensors. We will randomly select 50 volunteers of all ages, gender, and races from our local hospital and ask to first take their pulse and temperature with a commercial infrared thermometer and heart rate monitor and then with our Mtach. Similarly, we will acquire a mannequin and attach a commercial G-force sensor to it as we simulate it falling in a bathtub or down the stairs. Then, we will attach Mtach to the same mannequin and record the G-force as it falls in similar situations. Doing so will allow us to determine and improve the accuracy of our products.

Mdoc

The Mdoc prototype that we created in the past three weeks seeks to offer a proof-of-concept design that addresses two out of the three prioritized constraints that we answered for Question C. Currently, the Mdoc is hosted by a secure and easily accessible online database that anyone with the proper security clearance can view and the Mdoc can automatically detect and diagnose heart failures in patients who wear an Mtach device. We are currently hosting our database on PushingBox and hope to transfer data storage to our own database in the future.

Current Prototype:

Pass/Fail Criteria: If the Mdoc can correctly detect and diagnose heart failure, it passes. If not, it fails.

Material:
- Arduino
- PushingBox
- ThingSpeak

Stepwise Procedure:
1. Ask a person with normal heartbeat (60-80 BPM) to place his or her finger onto the Mtach so that the device sends a continuous stream of heartbeat data to Mdoc through the ESP Wifi module.
2. As the Mdoc collects the data and inputs them into Google Sheets, it will automatically calculate the wearer's heartbeat by calculating the frequency of the cardio waves.
3. Place a cool object (e.g. pencil, book, metal rod) onto the infrared heartbeat sensor. In approximately four seconds, the Mdoc will detect the wearer's aberrant heartbeat and send out an emergency signal to local EMS and the nearest Mrotor.

Future prototype:

In the near future, we hope to collect more heartbeat data from patients with a history of heart attacks and cardiac failure so that we can develop mathematical algorithms to make our Mdoc detection and diagnosis system more rapid and more accurate. For the temperature sensor, we will program a minimum and maximum temperature value (between 96.0F and 102.0F) into the sensor so that if the wearer's body temperature falls outside of these boundaries, then emergency medical personnel will be notified immediately. Finally, for the fall-detection data collection, we will strap our sensor to a testing dummy and record the changes in G-force as the dummy falls in a bathtub or down the stairs. We will then create an algorithm using the dummy data and upload that onto our Mdoc database so that if the Mdoc detects similar patterns in the future, it will understand to notify emergency personnel.

Mrotor

Because we were unable to acquire or build a full size quadrotor for this phase of prototyping, we relied on figures and data provided by the quadrotor company's website for logistical purposes. When we do purchase the quadrotor, we will be testing out its features to cross-reference them with the ones on the website. Since the quadrotor will be flying in outdoors conditions when it delivers the payload, we will test all specifications outside on the nearby school track. We will have the quadrotor fly a horizontal distance of 100 meters at a constant altitude and time the flight in order to determine its top speed. We will place weights in a box that's similar to the one that carries the automated external defibrillator fly the quadrotor around at a constant altitude to determine the quadrotor's sustainable flight time and sustainable payload weight. Finally, we will fly the quadrotor from the track to the school's baseball field and command the quadrotor to fly back on its own using its homing system to determine how accurate the GPS system on the quadrotor really is.

PORTFOLIO ELEMENT I

TESTING, DATA COLLECTION AND ANALYSIS

Mtach

In Part C (Presentation and Justification of Solution Design Requirements) of our engineering portfolio, we mentioned that our design constraints for Mtach were, in order of importance: inexpensive, non-invasive, lightweight, and energy efficient. In three weeks, we were able to develop a sensor that costs less than $15 to develop, is completely non-invasive, weights less than 300 grams, and uses very little battery power to detect a patient's heartbeat and transmit the data onto our online database. After we constructed our noninvasive heart beat monitor using an infrared sensor and an Arduino Microcontroller module, we ran many trials by detecting our own heartbeat and that of our parents and friends using our device. The results were quite accurate compared to the readings provided by a standard medical heart beat detector. The Mtach is capable of constructing an EKG of the wearer's heartbeat using the data that it collected.

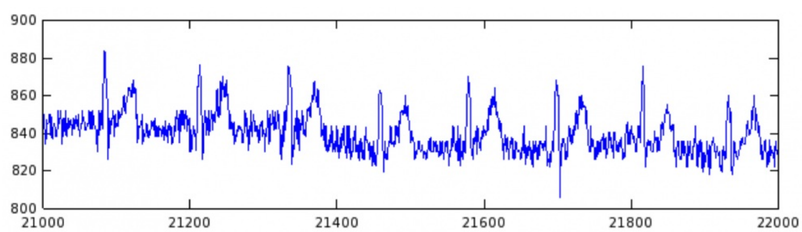


Figure I-1: The EKG diagram above was created using 1,000 values captured at 100Hz over a period of 10 seconds. This is just a snippet of the 600,000 values that was captured during the 16 minutes that that infrared heartbeat sensor was on. The EKG clearly shows the P waves, QRS complex, and T waves. If the wearer were to suffer cardiac failure, then the amplitude of the wave would decrease from 60 (820 to 880) to approximately 20 (840-860).

This allows us to approximate a person's heartbeats per minute by dividing the number of wave crests (e.g. local maximum of each heartbeat) by sixty. We hope to develop algorithms that can determine a person's systolic and diastolic pressure using the same data. In the future, we will also add more sensors to the Arduino module that will allow us to detect the patient's body temperature and whether that patient has fallen.

For data values used to create the EKG above, please see Appendix I-1.

Mdoc

Our prioritized list of constraints for the Mdoc were that the Mdoc had to automatically diagnose the medical emergency, automatically dispatch EMS personnel, and provide a secure and easily accessible database for medical professionals worldwide. Through our prototyping, we have created an Mdoc system that meets two out of three of those demands. Our heartbeat sensor can automatically detect when a person has gone into cardiac arrest with 95% false-positive accuracy rate. When the Mdoc detects a steady heartbeat from the Mtach, it will flash a green light at the crest and a black light at the trough of the heartbeat wave. However, if the Mdoc does not detect data input from the Mtach, a white light flashes on the screen and a signal is sent to the nearest emergency medical dispatcher.

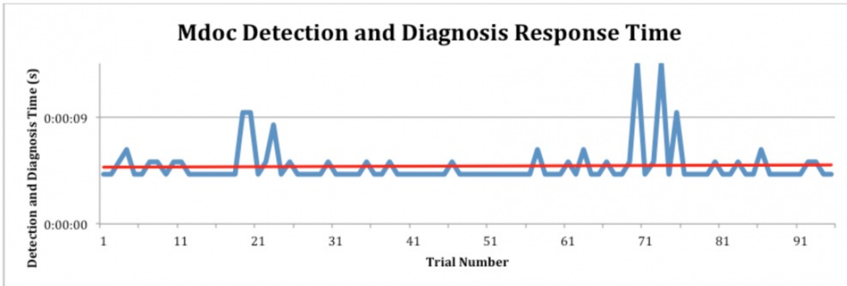


Figure I-2: The line graph above shows the amount of time it took for the Mdoc to detect and diagnose a heart failure. The red line that runs horizontally across the graph shows the median time it took for the Mdoc to respond. Our data analysis shows that it took approximately 4 seconds for the Mdoc to detect and correctly diagnose the heart attack.

For data values used to create the graph above, please see Appendix I-2.

Mrotor

Due to our limited budget, our main objectives during this phase of prototyping were to focus on creating a functional Mtach device and an operational Mdoc online database. We acknowledge the keystone role the Mrotor plays in making our iMrs vision a truly successful and feasible alternative to traditional fast-response vehicles. However, due to our lack of funding, we decided that it was in our best interest to postpone Mrotor development until after we receive a grant that would allow us to do R&D. In the meantime, we have looked into the work done by Dr. Vijay Kumar, Dean and Professor at the School of Engineering and Applied Sciences at the University of Pennsylvania. Specifically, we focused on his work relating to precision Autonomous Micro-UAV flight navigation in indoors and outdoors environments. Dr. Kumar's work could potentially have large and fitting applications for our Mrotor development because we can use his work to help us develop algorithms that help our quadrotor avoid flying into obstacles such as trees, telephone poles, and buildings.

Further Information:
Please visit Team iMrs's YouTube Channel here:
https://www.youtube.com/channel/UC-Suf5UpifzCYTmWP2f5n_g

- To view a computer animated rendering of how our product functions, please watch "iMrs Demo".
- To view our two-minute product introduction, please watch "Immediate Medical Response System (iMrs)".
- To view a real-life demonstration of how our product works, please watch "iMrs vs. Ambulance".
- To learn more about the technology behind our system, please watch "Prototype Technology Demo".

Appendix

831	827	850	833	827	830	836
831	836	850	829	830	830	833
826	833	854	836	836	831	829
831	837	850	830	833	826	836
838	838	854	826	838	829	830
860	831	847	830	830	833	826
859	834	852	827	826	836	830
868	838	852	830	831	836	827
854	839	842	832	826	834	831
854	841	838	830	829	827	833
838	836	838	830	833	830	831
818	847	834	827	836	830	844
819	842	838	827	836	836	839
836	838	833	830	830	833	855
833	842	828	836	830	839	862
829	842	836	833	830	844	862
836	847	826	836	830	838	864
830	843	827	836	833	827	857
830	860	827	830	836	838	853
827	854	831	830	838	830	828

Appendix I-1: The values above were use to create the EKG diagram (Figure I-1). These values were collected over the span of 1.4 seconds at 100 Hz, so they capture a full heartbeat from the onset of the P waves to the QRS complex to the T waves. Although we only show 140 values here, we captured over 600,000 values at the same frequency over a period of 16 minutes.

0:00:04	0:00:04	0:00:04	0:00:05	0:00:04
0:00:04	0:00:05	0:00:04	0:00:04	0:00:04
0:00:05	0:00:08	0:00:04	0:00:06	0:00:05
0:00:06	0:00:04	0:00:04	0:00:04	0:00:04
0:00:04	0:00:05	0:00:04	0:00:04	0:00:04
0:00:04	0:00:04	0:00:05	0:00:05	0:00:06
0:00:05	0:00:04	0:00:04	0:00:04	0:00:04
0:00:05	0:00:04	0:00:04	0:00:04	0:00:04
0:00:04	0:00:04	0:00:04	0:00:05	0:00:04
0:00:05	0:00:05	0:00:04	0:00:13	0:00:04
0:00:05	0:00:04	0:00:04	0:00:04	0:00:04
0:00:04	0:00:04	0:00:04	0:00:05	0:00:05
0:00:04	0:00:04	0:00:04	0:00:13	0:00:05
0:00:04	0:00:04	0:00:04	0:00:04	0:00:04
0:00:04	0:00:05	0:00:04	0:00:09	0:00:04
0:00:04	0:00:04	0:00:04	0:00:04	0:00:05
0:00:04	0:00:04	0:00:06	0:00:04	0:00:04
0:00:04	0:00:05	0:00:04	0:00:04	0:00:04
0:00:09	0:00:04	0:00:04	0:00:04	0:00:09

Appendix I-2: These values give the amount of time in seconds it took the Mdoc to detect and diagnose a medical emergency over 95 trials. The median time it took Mdoc to diagnose the emergency was 4 seconds, which is represented by the red line in Figure I-2 above.

References/Works Cited:

Kumar, Vijay. "Current Projects – Autonomous Micro UAVs." Vijay Kumar Lab. University of Pennsylvania, SEAS, n.d. Web. 15 Mar. 2015. <http://www.kumarrobotics.org/research/>.

"Data from ECG Recording." DIT Bioelectromagnetism. N.p., 28 Feb. 2014. Web. 15 Mar. 2015. <https://bioelectromagnetism.wordpress.com/2012/11/28/data-from-ecg-recording-in-todays-class/>.

PORTFOLIO ELEMENT J

DOCUMENTATION OF EXTERNAL EVALUATION

No content available

PORTFOLIO ELEMENT K

REFLECTION ON THE DESIGN PROJECT

No content available

PORTFOLIO ELEMENT L

PRESENTATION OF DESIGNER'S RECOMMENDATIONS

No content available