# Charlotte Area Robotics Technical Paper for the International Aerial Robotics Competition

Onkar Raut
*University of North Carolina at Charlotte, Masters Student EE*
Suraj Swami
*University of North Carolina at Charlotte, Masters Student EE*
Anthony Harris
*University of North Carolina at Charlotte, Undergraduate Student EE*
Malcolm Zapata
*University of North Carolina at Charlotte, PhD Student EE*

## ABSTRACT

This paper describes the mechanical, electrical, and software design in the making of a quadrotor for the 6th Mission of IARC. It is required to build a robot capable of navigating in an unknown environment, locating and retrieving a USB flash drive, avoiding detection by the facility's security infrastructure, and returning safely to the starting location. The solution chosen for this design was to implement basic flight control and balancing algorithms on the on-board microcontroller and off-loading "processor intensive" processes such as pattern recognition and image processing to a remote computer via a wireless link.

## INTRODUCTION

### Statement of the Problem

An autonomous aerial vehicle is needed to infiltrate a security compound in the Hesamic Republic of Nari, retrieve a USB flash drive and replace it with an identical flash drive while in a GPS denied environment. To accomplish this the vehicle must enter a 1m x 1m window and avoid being detected by a laser barrier. The robot cannot be larger than 1m x 1m x1m or weigh more than 1.5 Kg's.

### Conceptual Solution to Solve the Problem

This is to be completed by using a famous design known as a quadrotor made famous by De Bothezat back in 1923[11]. This particular design was chosen due to its stability, agility, and along with its ability to condense in size without losing any of the pre-stated qualities of the system. There is an onboard microcontroller on the quadrotor determining the relative speeds of the four motors as to keep the quadrotor balanced and in equilibrium, which will communicate with an intermediary microprocessor on the quadrotor between the microcontroller and the ground station. This was found to be the best way of communication so as to cut down on the latency of the flight control system within the microcontroller and adjustment of the motors.. Furthermore the microprocessor is capable of handling an onboard Robotic Operating System (ROS), to help with the communication process between the ground station, stereo camera pair, and under belly camera

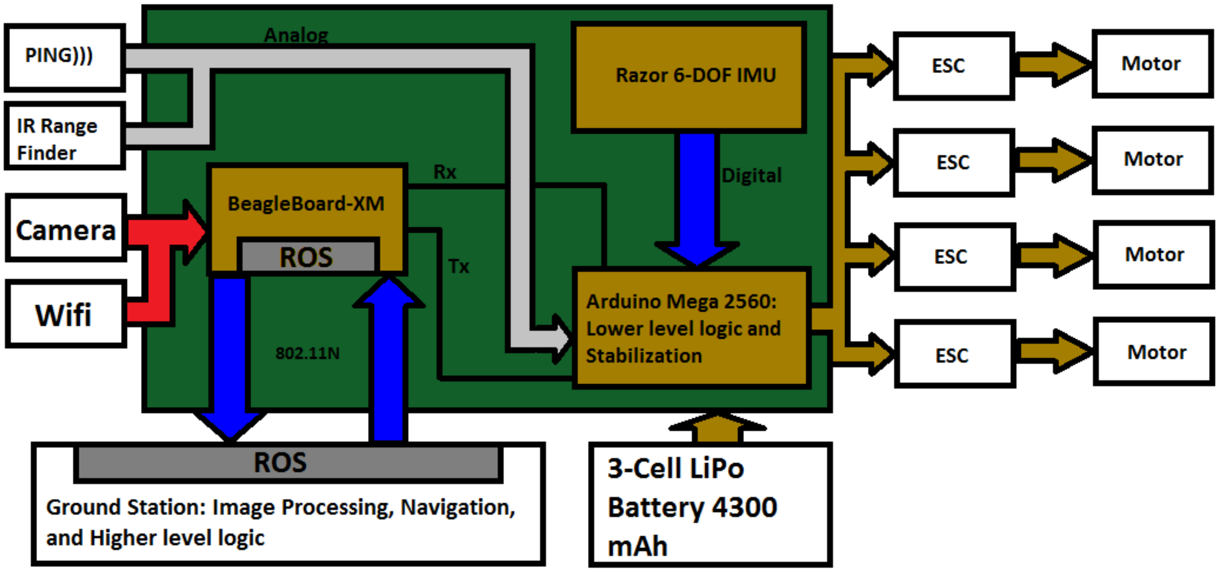


*Figure 1. Overall System Architecture*

## Yearly Milestones

With this being the Charlotte Area Robotics (C.A.R.) first attempt at creating a fully autonomous quadrotor to compete at the 6$^{th}$ mission IARC (International Aerial Robotics Competition), our Milestones are: the completion of design and build of a quadrotor frame; PCB board for distribution of power; Stereo pair used for stereoscopic vision; and the belly camera used as a way of determining the distance away from the object of interest.

## AIR VEHICLE

## Propulsion and Lift System

## Guidance, Navigation, and Control

### *Stability Augmentation System*

After preliminary research on different types of inertial measurement unit (IMU), our organization came to the decision on the Razor 6-Degrees of Freedom, SEN-10010 IMU. This particular model was selected due to its ability to take in readings about the gyroscope at a full scale of ±300 samples a second, while the accelerometer has a range of ±3g [8]. Furthermore built onto the board are the appropriate filters used to eliminate any unnecessary noise from interfering with the flight of the system.

The flight stabilization along with the movement is completed onboard allowing the system to automatically stabilize in the case of loss of communication with the ground station. This stabilization is thanks to a predefined open source code from AeroQuad[13] that has been modified to work with the IMU that was chosen along with the peripherals. AeroQuad code on The Arduino Mega 2560 was chosen due to its ability to interface with the additional peripheral sensors that are necessary for the system to operate as a fully autonomous robot.. From the open-source coding a PID Controller (Proportional Integral Derivative) was implemented in which the values for this particular controller were achieved through simulation and calibration. The proportionality value was found first in order to obtain a semi-stable quadrotor able to hover with slight oscillation. From this it was possible to calibrate the controller built into AeroQuad's.

The accelerometer data along with the gyroscope data about all three axes will be taken into contexts, allowing the quadrotor to know its attitude along with its distance traveled at any point in time. This data was gathered thanks again to AeroQuad's coding and its ability to relay the information from the IMU to the ground station every few milliseconds in a workable matrix format. Accompanying the dead reckoning calculations of the height of the system, a separate sensor, the **PING))) ultrasonic** sensor was added to the open-source coding as a verification tool on the accuracy of the system's altitude during flight.

### *Exploration and Navigation*

In order to explore the unknown environment, identify the correct "room", and find and retrieve the USB flash drive it important for the quadrotor to calculate its relative location in the environment. It was decided that a 3-D vision/ranging system paired with a mapping algorithm was the optimal choice for navigation in an unknown environment. Front-facing stereo cameras were used to provide 3-D images of the environment, which were then used in a SLAM algorithm (Simultaneous Localization and Mapping). This algorithm uses a combination of 3-D and peripheral sensors to "build up a map within an unknown environment (without a priori knowledge)… while at the same time keeping track of [its] current location". [2]

In addition to navigation, the quadrotor has to perform additional task such as detecting the window, laser switch, sign post and USB flash drive . These tasks of object recognition are performed as additional processes called "nodes" in ROS. A brief overview of the different object recognition algorithms is mentioned below.

The quadrotor executes a simple state machine of goals. Once a goal is achieved it changes its state to the next goal. A possible sequence of states could be set as: detect window; detect sign post; detect USB flash drive; retrieve USB flash drive; and calculate best return path. Best return path algorithm is implement using a landmark database created in the visual slam algorithm.

Visual Simultaneous Localization and Mapping (vSLAM) is implemented similar to SLAM[5]. vSLAM describes a robust localization and mapping algorithm using a single camera on a mobile robot. For our system the vSLAM[5] is further enhanced using a stereo camera for depth calculation to handle the 6DOF of a quadrotor.

vSLAM[5] implements SLAM using a landmark database of visual landmarks. A visual landmark is a collection of unique feature points extracted from an image using the SIFT algorithm (Scale-Invariant Feature Transform). It builds a map of environment and localizes itself in the map using these visual landmarks. vSLAM[5] performs visual measurement i.e. displacement of the robot with respect to certain landmark using visual odometry. For this system, visual odometry information is obtained from stereo camera images to estimate the distance traveled. In order to implement visual odometry, feature matching is performed using the SIFT algorithm on both left and right image. Normally SIFT returns a large number of features so an additional algorithm is implemented to find the best set of matching features RANSAC (RANdom Sample Consensus)[10] "is an iterative method to estimate parameters of a mathematical model from a set of observed data which contains outliers" [3]. . Triangulation is performed on the best matching features to calculate the 3D location of those points of interest. Similar operations are performed on the next pair of stereo images after a known interval of time. The SIFT algorithm is implemented to detect matching points between the consecutive pair of stereo images. Using the location of the matching points on the two pair of stereo images, it is possible to calculate the displacement, and thus location, of the robot.

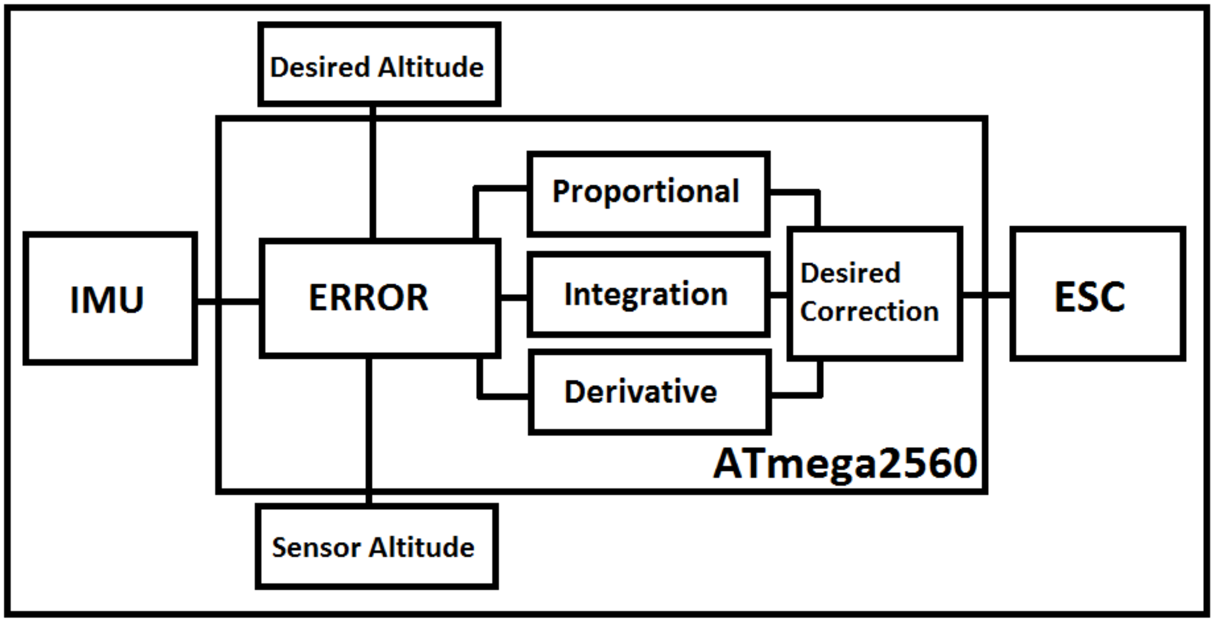


*Figure 2. Simplified PID Logic Flow Diagram*

## Flight Termination System

There are a few states in which the system is designed to terminate its current operation as safety precautionary for the platform and surroundings. The worst of the three conditions is the loss of signal between the microprocessor and the ground station. The other two are excessive oscillation, and user termination. In each of these cases a different subroutine will be called in order too avoid the least amount of damage to surrounding apertures and the platform. In the first case, loss of signal with the ground station, the microprocessor will execute a predetermined plan of first cancelling the current operation and attempt to reconnect with the ground station.. Once the connection attempt has been timed out, the quadrotor will descend at a steady state onto whatever surface is below it. While the excessive oscillation and user termination conditions take a much greater toll on the platform itself so as to avoid further catastrophic flaws. These conditions terminate power to the ESC's (Electronic Speed Controller), which are providing the power for each motor individually. With this condition enabled it causes the quadrotor to plummet towards the ground. Even though some of these termination systems can be costly to the platform, these precautions must be taken into consideration as to avoid unpredictable damage to the quadrotor and surrounding environment.

## PAYLOAD

### Sensor Suite

The overall purpose of the Sensor suite is to provide accurate data to the on-board microcontroller to perform the basic operation of stabilization of the platform and also provide high-level information to the SLAM algorithm as well as the object recognition algorithms. . Each of the sensors available on the platform are classified and described in the following sections:

### GNC Sensors

Control of stability of the platform is performed by the ATmega2560 microcontroller on-board the platform that directly reads the attitude of the quadrotor using a 6DOF Inertial Measurement Unit (IMU) comprised of the ADXL335 accelerometer, LPR530AL and LY530ALH MEMS gyroscope pair. The code for stability on board is a modified version of Aeroquad's v1.7 suited to work with our selected gyroscope. Transmitting serial commands to the ATmega2560 microcontroller from the microprocessor unit, which receives guidance commands wirelessly from a ground base, performs navigation. A calibrated pair of stereo cameras mounted on the forward side of the platform provides separate images for estimating the motion of the quadrotor in correspondence with values estimated from the IMU and PING))) sensor facing ground to accurately determine the current elevation of the platform.

.

*Figure 3. Razor 6-DOF IMU*


*Figure 4. Parallax Ping))) Sensor*

*Figure 5. Compact USB PC Webcam*

### Mission Sensors

For identifying the target, which is a non-occluded USB flash drive, an overhead camera is mounted to the bottom of the platform. This camera, similar to the frontal stereo camera pair, broadcasts the frames acquired to the ground base station for detecting the presence of an object of interest. It should also be noted that the stereo camera pair at the front of the platform can also be considered as mission specific sensors since they also provide the images for investigating the presence of a sign post on a wall.

#### *Target Identification*

Identifying the position of the object of interest (USB flash drive) is assisted by directing the quadrotor to the room that possibly contains the object of interest. The quadrotor performs a search algorithm looking out for signposts on walls that provide information using the frontal stereo camera pair. The under-belly camera mounted to the bottom of the quadrotor performs the search for the USB flash drive once in the room that is perceived to contain the USB flash drive. Each of the above two tasks are described below.

### *Detection of signposts*

A node in ROS (via OpenCV) is created which constantly broadcasts frames acquired from the on board frontal camera pair. Segmentation based on color of the object is performed on each frame acquired. The following algorithm is implemented:

1.) If an object of a desired color is found within the image, it is cropped from the image and sent for further processing.
2.) The cropped area of interest is converted to gray scale and boundary extraction is performed. Using the co-ordinates of boundary points extracted, a least squares ellipse is fit to the data.
3.) The lengths of the major and minor axes of the ellipse form a two dimensional feature vector input to a SVM classifier(Support Vector Machine) with a linear kernel trained with the dimensions of the three compound signs.

The SVM outputs a positive result depending upon the state the quad rotor is in and the type of signpost it is looking for. A similar algorithm is used to detect the laser barrier. Results using this are found in [7]

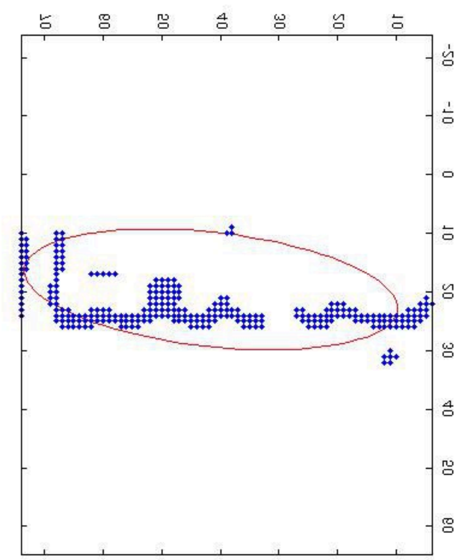

*Figure 6. MATLAB rendering of Detected Signpost*

### *Detection of USB thumb drive*

Another node in ROS (via OpenCV) is created for detecting the USB thumb drive. This node broadcasts the images acquired from the under-belly camera mounted at the bottom of the quad rotor. Gray-scale images acquired from this camera are transformed to remove the effects of projective transformation due to the attitude of the quad rotor. Segmentation i.e. thresholding is performed on each image by looking for dark groups of pixels. Each dark area is then clustered and shape recognition is performed using Fourier descriptors to relate the similarity to the desired USB thumb drive shape. The node returns a positive value, if a USB thumb drive is detected within the image, along with the pixel coordinates of the centroid of the USB thumb drive which can be transformed to input coordinates to the navigation system.

### *Threat Avoidance*

Five major threats have been identified as per our design and plan of action for retrieval of the object of interest. Three of these are intrinsic to the design of the quadrotor while the two others are a result of the environment. The three intrinsic threats to the mission are: collision with an indoor object, drastic loss of attitude and loss of communication. Collision with indoor objects is avoided by monitoring the four Infrared Range (IR) sensors present on the four sides of the platform. If collision is predicted, with the sensors reading less than half the entire range, the microcontroller increases the values of the two rotors in direction of the infrared sensor for a fixed duration of time or until the range on the IR sensor increases to more than $3/4^{th}$ of the entire range. If the range on the IR sensors falls below $1/4^{th}$ of the entire range; all motors are killed to avoid damage to the robot and the environment. A similar action is taken in case the second and third intrinsic threat condition occurs. Two other extrinsic threats to successful completion of the mission are the presence of the laser barrier and the security camera present at the entrance of the compound.

## Communications

Since the system is designed in two phases, the ground station and the actual maneuverable platform, data must be broadcast between the two. This is done in hardware by means of a wireless dongle, the ASUS WL-167G that is a WPA supplicant device with the driver available for the Beagleboard xM running a Linux operating system.

On the software side of communication, from the various available open source robotics frameworks for inter-process communication, such as Player[1] , MRPT [2], CARMEN [3] and Robot Operating System (ROS) [4], we choose ROS for this project. ROS (Robot Operating System) provides libraries and tools to help software developers create robotic applications. It provides hardware abstraction, device drivers, libraries, visualizers, message passing, package management, and more. ROS helped us to speed up the development process.

ROS is designed to provide peer-to-peer configuration such that we can have a number of processes, potentially on a number of different hosts, connected at runtime in a peer-to-peer topology. This helped us to setup the communication between the beagle board on the quadrotor and the ground station. In additional to this, the ROS framework also provided an inter process communication between the different processes running different algorithm using common data such as images, IR data, etc. This was implemented using various functionalities provided by ROS such as nodes, messages, topics, and services. Since ROS is integrated with OpenCV, image-processing algorithms were written in OpenCV.

A node is where the main processing is performed. Different algorithms such as mapping, SLAM, feature detection, and USB flash drive detection would each form a node. A special node is written which acts as a device driver to control the quadrotor. This node runs on the beagle board. For communication among these nodes, each node sends a message by publishing it to a given topic, which is simply a character string such as "left_camera" or "IR_data". We have various nodes, for example, sign post detection, USB flash drive detection, feature detection, etc., which can subscribe to the "left_camera" and publish their output. A different node such as SLAM will subscribe to these outputs and make its decision. ROS helped to provide an inter-process communication between various modules in the quadrotor and simplified the wireless transfer of data from the platform to the ground station.

In addition to this, ROS also provided various packages such as SIFT library, stereo image publishing, etc., which further speeded up the development process.

*Figure 7. Robotic Operating System*

*Figure 8. Asus USB Network adapter*

## Power Management System

With the all the peripherals attached to the Arduino Mega 2560 board, a total consumption of the board was measured to be at a rate of one and a half amps an hour. With this in consideration the four ESC's would consume one amp an hour, which is used to power the Atmega2560 microprocessor, the Beagleboard-XM during flight along with the four brushless DC motors. Meanwhile the Beagleboard-XM fully loaded with the three vision cameras and wireless dongle by Asus consumes four and a half amps an hour. The total power consumption of the system was found to be ten amps an hour, which led to a 4000mAh battery that gives the system an average flight time of twenty-four minutes. Due to the many various components of the system, a custom PCB (Printed Circuit Board) was designed to help with distribution of power.

# OPERATIONS

## Flight Preparations

Before the motors of the system are capable of being enabled a few precautionary steps must be taken as to insure the safety of the system along with the surrounding environment.

- Ground station communication is declared operational through the Beagleboard-XM
- Serial Communication is declared operational between the ATmega2560 and the Beagleboard-XM
- Infrared range finders are online
- Ultrasonic range finder is online for height stabilization
- Enable the four motors by applying power to the ESC's

## Man – Machine Interface

Tools available from ROS such as rosbag, rxgraph, rostopic and rviz are the readily available candidates to assert the state of the quadrotor and the items being processed. As per the definition on the wiki page, rxgraph is a command-line tool for visualizing a ROS computation graph, rxbag is a GUI tool for visualizing, inspecting and replaying histories of ROS messages, rostopic contains the rostopic command-line tool for displaying debug information about ROS Topics, including publishers, subscribers, publishing rate, and ROS Messages, whereas, rviz is a 3-D visualization environment for robots using ROS. These help us visually interpret what the robot is trying to do autonomously. All sensors on board the platform are nodes, which publish messages in topics to which the processing nodes subscribe from the ground station computer that is also running ROS. This makes for the perfect environment to run all the above-mentioned tools to monitor the man-machine interface.

## RISK REDUCTION

### Vehicle Status

The vehicle status is determined by a master node on the ground station which runs the state machine by looking at publications in multiple topics such as the message board containing results of USB flash drive detection node, signpost recognition algorithm, stabilization, etc. Monitoring this master node would constantly convey the status of the vehicle to the observer.

### *Shock – Vibration Isolation*

The worst effect of a severe fall to the quadrotor is damage to the IMU. Since each IMU is unique, the entire system may require recalibration, including tuning the PID controller. Although the 6 degree of freedom IMU is said to survive 3000g acceleration for a small period of time (0.5s), in order to reduce shock of a unsafe landing high dampening springs are added to the mechanical design along with high density foam. This also assists reduce vibrations during flight. The analog output of the IMU is also low pass filtered by an analog filter before being passed to the analog to digital converter of the microcontroller.

### *Protection against Electromagnetic and Radio Frequency Interference*

The wireless adapter used to connect wirelessly to the maneuvering platform is the ASUS WL-167G. While designing the WL-167g, it has been mentioned in the product and company description that ASUS engineers paid special attention to EMI (electromagnetic interference), thermal, acoustics and details that usually go unnoticed to achieve complete customer satisfaction.

### *Safety*

While results and research are important, more important is the safety of the operator and the robot. The following guidelines ensure the safety of both:

- Safety Glasses must be worn at all times when robot is activated.
- Operator and all human personnel must stay behind a net during robot operation.
- Testing of the quadrotor should be performed on soft ground such as grass turfs to minimize damage to the robot incase of operational mishaps.
- The quadrotor may not be operated at heights more than 6 feet and 100 feet radial distance from the ground station.
- The quadrotor should be tethered to the ground when performing calibration, PID tuning or testing new flight algorithms till satisfactory flight results are achieved.

### *Modeling and Simulation*

Testing of the individual algorithms for object recognition were initially performed in MATLAB and then translated into OpenCV code to enable ROS. The model of the quadrotor and simulation of the mechanical design, geometrical properties, material strengths and physics were simulated in SolidWorks.

### *Testing*

Testing is one of the most important and potentially most dangerous phases in any engineering design process. The safety guidelines mentioned above were followed. Initial tests to calculate the values of the PID controller were obtained while the vehicle was being held by one of the software designers. Further testing was done using tethers attached to the quadrotor prevent a catastrophic vehicle collision. It was possible to read values of the various sensors as the quadrotor was flying in order to gain immediate feedback about how the system should behave. This quick feedback loop from testing to programming and design is very important toward making progress.

## CONCLUSION

An autonomous aerial system is a challenging engineering problem requiring the integration of many disciplines of engineering. Software, electronics, and mechanics must all work in synchrony to provide a stable and balanced system. While some of the design decisions seemed obvious and used off-the-shelf components and software libraries, much of the design was very open-ended and required creative solutions to problems that do not have very good previously-defined solutions. Weight of the mechanical and electrical components as well as the battery used to make the quadrotor autonomous were a major design concern and were considered in every aspect of the autonomous quadrotor's design. The solution implemented on this quadrotor communicated with a remote PC to perform much of the processing, but future systems will aim to take on more and more of the processing on board the quadrotor, eliminating the need for external control.

## REFERENCES

[1]http://playerstage.sourceforge.net/.

[2]http://www.mrpt.org/

[3]http://www.cs.cmu.edu/~carmen/links.html

[4]www.ros.org

[5]Niklas Karlsson, et. al. The vSLAM Algorithm for Robust Localization and Mapping.
IEEE International Conference on Robotics and Automation Barcelona, Spain, April 2005

[6]David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110 2004.

[7]Survey of Recognition of Arabic Scripts for Indoor nmanned Aerial Vehicles using Classical Methods for pattern Recognition, IEEE Southeast conference 2011. IEEE DOI: 10.1109/SECON.2011.5752944

[8] http://www.sparkfun.com/datasheets/Sensors/IMU/lpr530al.pdf

[9] Woo Yeon Jeong; Kyoung Mu Lee; , "Visual SLAM with Line and Corner Features," Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on , vol., no., pp.2570-2575, 9-15 Oct. 2006

[10] Martin A. Fischler and Robert C. Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, 6 (June 1981), 381-395. DOI=10.1145/358669.358692
http://doi.acm.org/10.1145/358669.358692

[11]de Bothezat, George. The General Theory of Blade Screws. NACA Report 29, 1920

[12] David G. Lowe, "Object Recognition from Local Scale-Invariant Features," Computer Vision, IEEE International Conference on, p. 1150, Seventh International Conference on Computer Vision (ICCV'99) - Volume 2, 1999