MRSL: AUTONOMOUS NEURAL NETWORK-BASED SELF-STABILIZING SYSTEM

By

Hooman Hedayati

December, 2015

Director of Thesis: Dr. Nasseh Tabrizi

Major Department: Computer Science

Stabilizing and localizing the positioning systems autonomously in the areas without GPS accessibility is a difficult task. In this thesis we describe a methodology called Most Reliable Straight Line (MRSL) for stabilizing and positioning camera-based objects in 3-D space. The camera-captured images are used to identify easy-to-track points “interesting points” and track them on two consecutive images. The distance between each of interesting points on the two consecutive images are compared and one with the maximum length is assigned to MRSL, which is used to indicate the deviation from the original position. To correct this our trained algorithm is deployed to reduce the deviation by issuing relevant commands, this action is repeated until MRSL converges to zero. To test the accuracy and robustness, the algorithm was deployed to control positioning of a Quadcopter. It was demonstrated that the Quadcopter (a) was highly robust to any external forces, (b) can fly even if the Quadcopter experiences loss of engine, (c) can fly smoothly and positions itself on a desired location.

MRSL: AUTONOMOUS NEURAL NETWORK-BASED SELF-STABILIZING SYSTEM

A Thesis

Presented To the Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Software Engineering

By

Hooman Hedayati

December, 2015

MRSL: AUTONOMOUS NEURAL NETWORK-BASED SELF-STABILIZING SYSTEM

By

Hooman Hedayati

APPROVED BY:

DIRECTOR OF THESIS: ____

Nasseh Tabrizi, PhD

COMMITTEE MEMBER: ____

Venkat N. Gudivada, PhD

COMMITTEE MEMBER: ____

Gopalakrishnan, K., PhD

CHAIR OF THE DEPARTMENT OF COMPUTER SCIENCE: _________________________

Venkat N. Gudivada, PhD

DEAN OF THE GRADUATE SCHOOL:____

Paul J. Gemperline, PhD

DEDICATION

To dedicate my thesis to my beloved parents “Mohammad and Minou” for always supporting me, to my lovely sibiling “Mahsa and Reza” for showing me the right path in life and to Dr. Tabrizi for giving me a chance to prove myself.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Dr. Nasseh Tabrizi, for the continuous support of my thesis and related research, for his patience, motivation, and immense knowledge. His guidance helped me throughout my research and writing of this thesis. I could not have imagined having a better advisor and mentor for my thesis study.

Evaluation of the reliability of MRSL was partially supported by NSF REU grant with undergraduate student participation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF SYMBOLS

N	Newton
cm	centimeter

LIST OF ABBREVIATIONS

Miniature Aerial Vehicles (MAVs)

Inertial Measurement Unit (IMU)

Global Positioning System (GPS)

Simultaneous localization and mapping (SLAM)

SPLAM (simultaneous planning, localization and mapping)

LRF (Laser Ranging Finder)

EKF(Extended Kalman Filter)

PTAM(Parallel Tracking and Mapping)

Most Reliable Straight Line (MRSL)

Sum of Squared Differences (SSD)

CHAPTER 1 – INTRODUCTION

The demands for utilizing autonomous vehicles in hazardous situations are increasing rapidly. There are plenty of examples where these vehicles are successfully used in practice, such as those inspecting the damaged nuclear reactors in Fukushima [1], robotic submarines attempting to repair the ruptured well in the Gulf of Mexico [2] and firefighters rescuing survivors in a collapsing building [3]. Among these robots, flying Miniature Aerial Vehicles (MAVs) have become an important tool not only in the military domain, but also in civilian environments. Particularly Quadcopters are becoming more popular, especially for observational and exploration purposes in indoor and outdoor environments, but also for data collection, object manipulation or simply as high-tech toys. Also, Quadcopters have many more potential applications such as: A swarm of small, light and cheap Quadcopters could for example be deployed to quickly and without risking human lives explore collapsed buildings to find survivors. Equipped with high-resolution cameras, MAVs they can also be used as flying photographers, providing aerial based videos of sport events or simply taking holiday photos from a whole new perspective [4].

Among all MAVs, Quadcopters are becoming more popular for observational and exploration purposes in indoor and outdoor environments. Quadcopters have flying behavior similar to a helicopter, which means they can fly and land vertically (unlike airplanes), stay perfectly in a certain point in space (like helicopters) and move in any given direction at any time, without having to turn first. With these features, Quadcopters can maneuver extremely well in constrained high-dense indoor spaces, as corridors or offices, making them ideally suited for stationary observation or exploration in obstacle-dense or indoor environments, making them one of the best choices for indoor environments exploration compared to other robots [4].

In order to navigate, Quadcopters often rely on a wide variety of sensors including Inertial Measurement Unit (IMU), Global Positioning System (GPS), and gyroscopes. Although these sensors are not completely reliable individually, combinations of them arguably make them good enough as outdoor positioning systems. However, flying in an unknown environment without GPS signal requires alternative positioning methods that include expensive sensors like a 3-D depth scanning camera [5]. Alternatively, one can use optical cameras to collect information [6]. While cameras are cost efficient, they have some disadvantages where excessive amounts of data is collected which in turn makes processing data computationally expensive. In addition, 2-D images that are captured using cameras make it difficult to extract 3-D information.

One crucial objective of any Quadcopter is to localize and stabilize itself by maintaining its position by constantly counteracting minor randomly induced movements. Although IMUs help to achieve this, but a major disadvantage of using IMUs for navigation is that they typically suffer from accumulated errors, including the “Abbe Error” [7], which describes the magnification of angular error over distance. Furthermore, as stabilizing systems continually add detected changes to its previously calculated positions; any errors in measurement are accumulated leading to 'drift', or an ever-increasing deviation from the actual location, making IMUs not reliable [7].

The task of accurate localizing of a Quadcopters in previously unseen environments is widely investigated [8]. This thesis presents an innovative approach for stabilizing camera-based objects in 3-D. Although accuracy and robustness of our approach is tested on a Quadcopter, this algorithm can be implemented on any 3-D positioning systems.

The objective of this thesis is to develop a system capable of controlling and self-stabilizing of the Quadcopter in a previously unknown environment using only frontal camera in order to compute an absolute estimate of the Quadcopter’s pose by applying visual tracking methods. This pose estimate can then be used to calculate the control commands required to fly to and hold a desired position in 3-D space.

The thesis report will unfold as follows –

- Chapter 2 will overview of the related work
- Chapter 3 will introduce the Quadcopter used (the Parrot AR.Drone) and state its capabilities and available sensors. Also will describe briefly, how the Quadcopter fly.
- Chapter 4 will explain the methodology called “Most Reliable Straight Line”.
- Chapter 5 will introduce the Coordinator.
- Chapter 6 will validate the methodology through the results produced.
- Chapter 7 will conclude our research.

CHAPTER 2 – RELATED WORK

The problem of accurately tracking the motion of a robot in an arbitrary, previously unseen environment has been the focus of a lot of research in the field of computer vision and robotics, and is widely known as the Simultaneous Localization and Mapping (SLAM) problem. It was originally developed by Hugh Durrant-Whyte and John J. Leonard [9] based on earlier work by Smith, Self and Cheeseman [10]. Durrant-Whyte and Leonard originally termed it SMAL but it was later changed to give a better impact. SLAM is concerned with the problem of building a map of an unknown environment by a mobile robot while at the same time navigating the environment using the map. SLAM consists of multiple parts; Landmark extraction, data association, state estimation, state update and landmark update.[11]

The general idea is very straight-forward: Using available sensor data, a map of the environment is generated. This map in turn is used to re-estimate the new position of the robot after a short period of time. A SLAM system thus aims at answering the two questions “What does the world look like?” and “Where am I?”. This process can furthermore be done actively, that is navigating a robot such that new information about the environment can be acquired while assuring that the current pose can still be tracked accurately. Such approaches are also called Simultaneous Planning, Localization and Mapping (SPLAM) [12].

While the SLAM and PSLAM literature is vast, only a small number of solutions facilitate online control and none of them combine the result of SLAM algorithm for self-stabilizing problem. Furthermore, Most of the SLAM literature focuses on active range sensing, such as the SICK laser range-finder. Some of them are as fallow:

Keyframe-Based SLAM: The early days, it used Laser Ranging Finder (LRF) and sonar sensors [13,14] but recently, vision sensor was also used [15,16]. The first stage of vision-based SLAM, only the sensor was substituted to a camera from LRF or sonar, keeping the overall framework of SLAM using probabilistic models such as EKF(Extended Kalman Filter) and particle filter. Davison et al.'s work [15] was based on EKF and constructed a map of 3-D sparse points so that it contained uncertainties on the camera's position by nature. Later, Parallel Tracking and Mapping (PTAM) so-called keyframe-based SLAM appeared [16]. To localize a moving camera in an unknown scene, tracking and mapping are separated and run in two parallel threads. PTAM constructs a map of 3-D points based on keyframes collected during tracking process using batch techniques, Bundle Adjustment and the map can contain thousands of points, which allows precise camera localization more proper for Augmented Reality applications [17].

Scene Modeling: The detection of higher level structure in a real world has been progressed using both visual and non-visual sensors. Planes were fitted to clouds of points reconstructed using LRF [18,19]. The discovered planes were, however, not incorporated into a map. Afterward, many have attempted to discover planes in vision-based SLAM as in [10,11,12]. Rachmielowski et al.'s work [20] attempted to get 3-D reconstruction of a scene during SLAM tracking. They connected sparse 3-D points to compose a mesh using a Delaunay triangulation and reconstructed a scene model with the meshes. Thus, the reconstruction is unlikely a good estimation of the real surface because the 3-D points used for a mesh were sparse and involved uncertainties by nature of the SLAM approach. Chekhlov et al.'s work [21] also tried 3-D reconstruction of a scene during SLAM tracking. However, to find planes they approached with RANSAC process instead of the Delaunay triangulation used in [20]. In view of using RANSAC process, our system is similar to their system but we go beyond their work by incorporating the discovered planes into the map and using them during the tracking process[17].

Also, J. J. Engel did a research on application of SLAM and implement the result of it on a Quadcopter. As a result of his work, enables a Quadcopter to accomplish tasks such as [4]:

- holding a flying position in spite of external disturbances and interference such as wind
- high-level manual control of the Quadcopter: Instead of directly piloting the drone, this system enables the pilot to send high-level control commands

In this thesis, our approach is to use Neural Network combination of SLAM in order to achieve self-stabilization.

CHAPTER 3 – QUADCOPTERS

The concept of an aircraft flying with four horizontally aligned rotors had already been proposed in 1922 [23], The de Bothezat helicopter, built by the US Army (Figure 1 [23]) It is considered to be one of the first successful helicopters ever built, however it only ever reached a height of approximately 9 m, and stayed in the air no more than 2:45 minutes. Because of two weak points this design quickly disappeared and was dominated by the much more common two-rotor helicopter. Original Quadcopters were not cost and energy efficient compare to helicopters. Second, Quadcopters were inherently unstable and hence difficult to control - without the help of advanced electronic control systems and stabilizing routines, manual control turned out to be too complex.

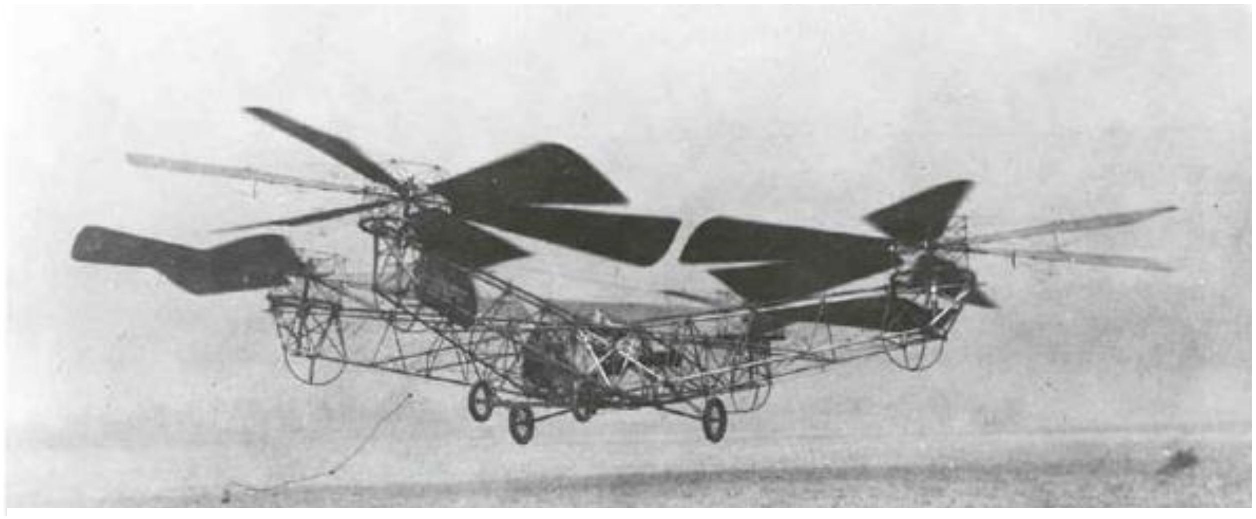

Figure 1: The de Bothezat helicopter, built in 1922 by the US Army.

With advancement in electrical engineering and growing importance of MAVs, development of new generation of Quadcopters increasingly becoming more popular again. Because it is less mechanically complex than a normal helicopter as all four rotors have a fixed pitch. Also, all rotors can be fenced by a frame, protecting them in collisions and permitting safe flights indoors and in obstacle-dense environments. Finally, the use of four rotors allows each to have a smaller diameter, causing them to store less kinetic energy during flight and reducing the damage when the rotor hit a surrounding object, making Quadcopters significantly safer to use close to people. [4]

The rise in popularity of the Quadcopters demands extensive research in the design to improve reliability and safety. The most important problem with the current design is “efficiency rate by distance. As airplanes fly longer distances, they burn fuel and as consequence the total weight of airplane should be reduced over time. In contrast, Quadcopters use batteries as power supply. Overtime, the batteries of the Quadcopter discharge, since we cannot get rid of them, the Quadcopter should carry unusable weight.

Modern definition of a Quadcopter, also called a quadrotor helicopter or quadrotor [24], is a multirotor helicopter that is lifted and propelled by four rotors. Quadcopters are classified as rotorcraft, as opposed to fixed-wing aircraft, because their lift is generated by a set of rotors (vertically oriented propellers). Quadcopters generally use two pairs of identical fixed pitched propellers; two clockwise (rotors M2 and M4) and two counter-clockwise (rotors M1 and M3). These use independent variation of the speed of each rotor to achieve control (Figure 2). By

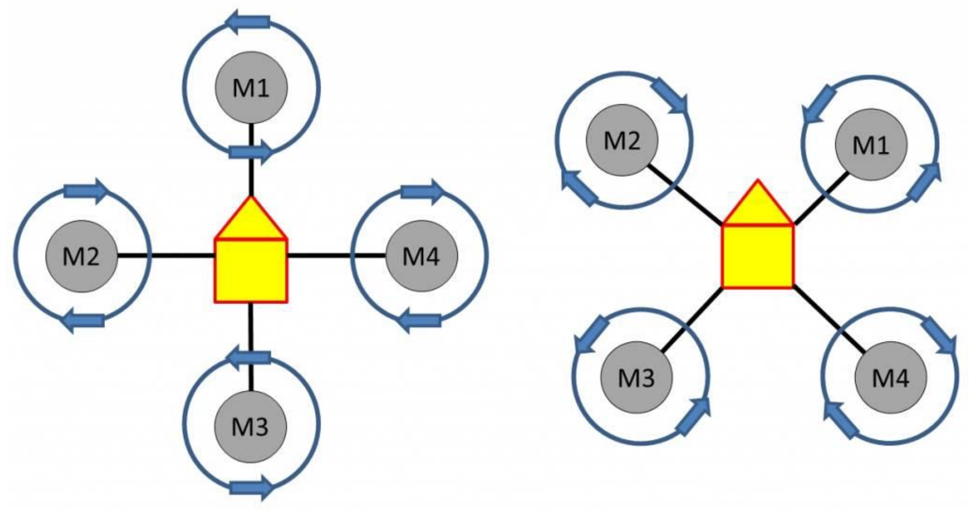

Figure 2: Quadcopter's rotors spin

changing the speed of each rotor it is possible to specifically generate a desired total thrust; to locate for the center of thrust both laterally and longitudinally; and to create a desired total torque, or turning force [25,26].

MAV cab flies freely in three dimensions; therefore three different types of motions are introduced. Imagine three lines running through an airplane and intersecting at right angles at the airplane's center of gravity as fallow: pitch, nose up or down about an axis running from wing to wing; yaw, nose left or right about an axis running up and down; and roll, rotation about an axis running from nose to tail (Figure 3). The axes are alternatively designated as lateral, vertical, and longitudinal. These axes move with the vehicle and rotate relative to the earth along with the craft. These definitions were analogously applied to spacecraft when the first manned spacecraft were designed in the late 1950s.

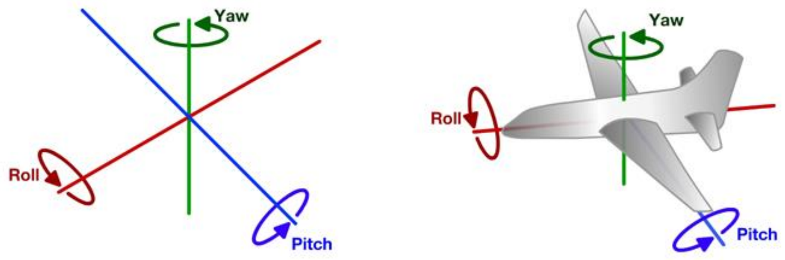

Figure 3: Roll, Pitch and Yaw

The same coordinates can be applied to the Quadcopters. So the following actions can be taken to maneuver the Quadcopter:

- vertical acceleration is achieved by increasing or decreasing the speed of all four rotors equally,

- Yaw rotation can be achieved by increasing the speed of engines 1 and 4, while decreasing the speed of engines 2 and 3 (or vice-versa) - resulting in an overall clockwise (or counter-clockwise) torque, without changing overall upwards thrust or balance,
- Horizontal movement can be achieved by increasing the speed of one engine, while decreasing the speed of the opposing one, resulting in a change of the roll or pitch angle, and thereby inducing horizontal acceleration.

Moreover, the combination of three main actions could be taken to maneuver the Quadcopter. As a final point there exist eight different maneuvers as shown below (Figure 4):

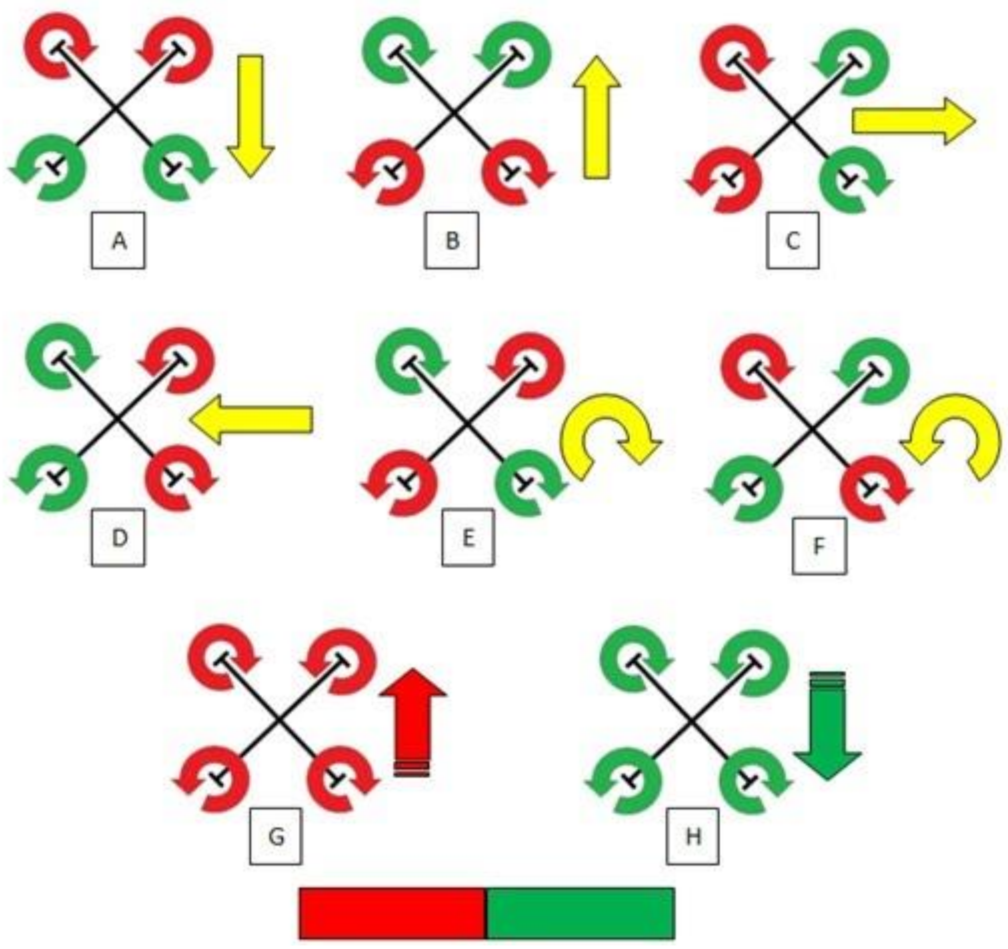

Figure 4 : Basic Movement of a Quadcopter

A. Pitch back
B. Pitch forward
C. Roll right
D. Roll left
E. Yaw clockwise
F. Yaw anti-clockwise
G. Ascend
H. Descend

To test the accuracy and robustness of the algorithm which will be explained in chapter 4, the algorithm was deployed to control positioning of a “Parrot AR.Drone”, which is a Quadcopter built by the French company Parrot [27]. The Parrot AR.Drone has dimensions of 52.5 cm × 51.5 cm with, and 45 cm× 29 cm without hull. It has four rotors with a 20 cm diameter, fastened to a robust carbon-fiber skeleton cross providing stability. A removable styrofoam hull protects the drone and particularly the rotors during indoor-flights, allowing the drone to survive minor and not-so-minor crashes such as flying into various types of room furniture, doors and walls – making it well suited for experimental flying and development. An alternative outdoor-hull -missing the rotor-protection and hence offering less protection against collisions - is also provided and allows for better maneuverability and higher speeds.

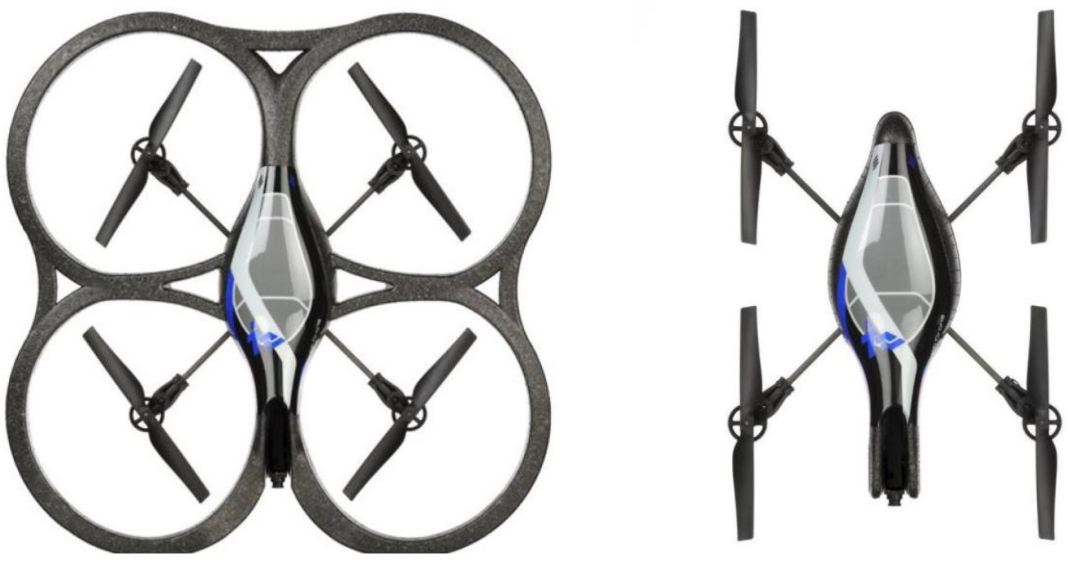

Figure 5 : Top view of Parrot AR.Drone

The drone weights 380g with the outdoor-hull, and 420 g with the indoor-hull. Although not officially supported, in our tests the Quadcopter was able to fly with an additional payload of up to 120 g using the indoor hull - stability, maneuverability and battery life however suffered significantly, making the drone hardly controllable with that kind of additional weight. The drone is equipped with two cameras (one directed forward and one directed downward), an ultrasound altimeter, a 3-axis accelerometer (measuring acceleration), a 2-axis gyroscope (measuring pitch and roll angle) and a one-axis yaw precision gyroscope. The onboard controller is composed of an ARM9 468MHz processor with 128Mb DDR Ram, on which a BusyBox based GNU/Linux distribution is running. It has an USB service port and is controlled via wireless LAN [27].

Software for Parrot AR.Drone is not open source and there is not good documentation for it. For this thesis we don’t need complex control on the Quadcopter so we assume it as a black box. There are four built-in communication channel available. Although the approach in this thesis is not using all channels, for the sake of completeness, all four channels would be introduced. As soon as the battery is connected, the drone sets up an ad-hoc wireless LAN network to which any device may connect. Upon network connection, the drone immediately starts to communicate (sending data and receiving navigational commands) on four separate channels[4,28]:

- **navigation channel (UDP port 5554)**

While in normal mode the Quadcopter only broadcasts basic navigational data every 30 ms, after switching to debug mode it starts sending large amounts of sensor measurements every 5 ms. The exact encoding of the sent values will not be discussed here, it is partially documented in [28]. The most important parameters and sensor values - and the ones used in our approach - are the following:

1) Quadcopter orientation as roll, pitch and yaw angles: as mentioned in the previous section, roll and pitch values are drift-free and very accurate, while the measured yaw-angle is subject to significant drift over time
2) Horizontal velocity: in order to enable the drone to keep its position in spite of wind, an optical-flow based motion estimation algorithm utilizing the full 60 fps from the floor camera is performed onboard, estimating the drone's horizontal speed. The exact way these values are determined however is not documented.

- **video channel (UDP port 5555)**

The drone continuously transmits one video stream, which can be one of four different channels - switching between channels can be accomplished by sending a control command to the drone. The four available channels are depicted in Figure 2.3. As can be seen, neither of the available cameras can be accessed fully: for the downwards facing camera the available frame rate is - with only 18 fps - significantly lower than the original 60 fps. Furthermore the maximal supported resolution is 320×240, halving the forward camera's original resolution1 [4,28].

- **command channel (UDP port 5556)**

The Drone is navigated by broadcasting a stream of command packages, each defining the following parameters:

1. desired roll and pitch angle, yaw rotational speed as well as vertical speed, each as fraction of the allowed maximum, i.e. as value between -1 and 1,

2. one bit switching between hover-mode (the drone tries to keep its position, ignoring any other control commands) and manual control mode,

3. one bit indicating whether the drone is supposed to enter or exit an error-state, immediately switching off all engines,

4. one bit indicating whether the drone is supposed to take off or land.

Being sent over an UDP channel, reception of any one command package cannot be guaranteed.

In our implementation the command is therefore re-sent approximately every

10 ms, allowing for smoothly controlling the drone [4,28].

- **control port (TCP port 5559, optional)**

Control commands can be used to change internal settings of the drone, for example for switching between the four available video channels. In general a control command is transmitted as a string of the format "[attribute]=[value]", for a list of the available commands we refer to [4,28].

CHAPTER 4 – METHODOLOGY

As it is shown in Figure 6, the camera captures images that are used to identify easy-to-track points named "Landmarks" and tracks them between two consecutive images. The longest distance between the landmarks on the two consecutive images is assigned to Most Reliable Straigh Line (MRSL), which is used to indicate the deviation from the Quadcopter's original position. Our trained algorithm is then deployed to reduce the deviation by issuing relevant commands. This action is repeated until MRSL converges to zero.

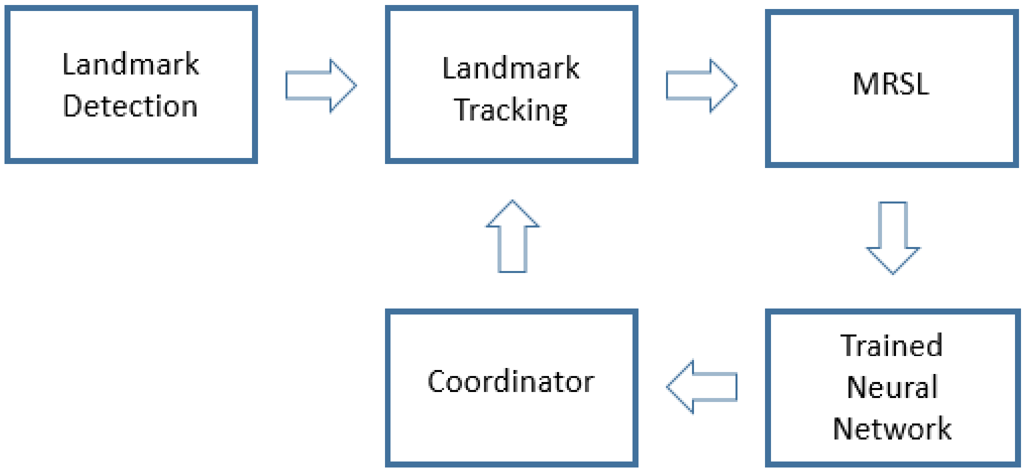

Figure 6: Methodology

3.1 Landmark Detection and Tracking

Movement of the camera, which implies to movement of Quadcopter, cause changes in the position of landmark points. The movements of these points are tracked in the next step by extracting the magnitude and direction of Quadcopter movement. Some of the features that can be extracted from the images are edges, corners, blobs, or ridges. Several feature detection algorithms like SIFT (DoG) [29], SURF [30], FAST [31] and Harris corner detector [32] were considered for this study but because the high-speed requirement, it was decided to adopt the Features From

Accelerated Segment Test (FAST) algorithm. FAST is a corner detection method, which can be used to extract landmarks by using a circle of 16 pixels to classify whether center pixel of the circle is aligned with a landmark. Decision is made by comparing color intensity of the center pixel with those of 16 neighboring pixels located on the perimeter of the circle. For each candidate pixel P (see Figure 7), the algorithm identifies which pixels are aligned with a circle perimeter: if a long enough sequence of continuously brighter or continuously darker pixels is found, it is classified as a landmark.

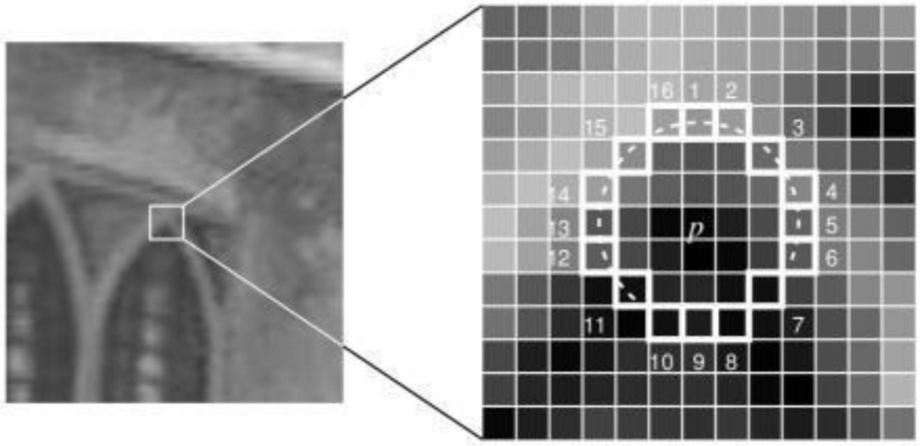

Figure 7 The FAST-16 corner detector

Once the landmarks are detected, the exact positions of them are extracted next. A general formula for tracking is to find parameters p of a warp function $\boldsymbol{f(x,y;p)} : \mathbb{R}^{\boldsymbol{2}} \times \mathbb{R}^{\boldsymbol{d}} \rightarrow \mathbb{R}^{\boldsymbol{2}}$, such that the difference between the original patch $\boldsymbol{T(x,y)}$ and the transformed image $\boldsymbol{I(f(x,y;p))}$ becomes minimal, that is minimizing the sum of squared differences (SSD) [4], see Eq. 2.

$$\boldsymbol{E_{SSD}(P)} \coloneqq \sum_{x,y} \left(\boldsymbol{I(f(x,y;p)) - T(x,y)}\right)^{\boldsymbol{2}} \qquad (1)$$

$$\boldsymbol{P^{*} = arg\ min\ E_{SSD}(p)} \qquad (2)$$

Choosing a good warp function is critical to having a suitable degree of freedom, "Pure Translation" function selected as a warp function, which tracks a two-dimensional image patches. The resulting transformation has two degrees of freedom, the displacement in two dimensions is shown in Eq. 3.

$$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\delta x},\boldsymbol{\delta y}) = \begin{pmatrix} \boldsymbol{x}+\boldsymbol{\delta x} \\ \boldsymbol{y}+\boldsymbol{\delta y} \end{pmatrix} \quad \textbf{(3)}$$

3.2 Extracting the Movements

Let C_1 and C_2 be two consecutive images captured by the camera. First FAST algorithm is deployed on C_1 to find all landmarks coordinates ($P_1 , P_2 , \dots , P_n$). Then tracking algorithm is run on C_2 to track same landmarks and find their new coordinates ($P_1' , P_2' , \dots , P_n'$). By connecting each pair P_m to P_m' a line is formed and labeled "Reliable Straight Line" (RSL). RSLs are important because they can be used to estimate camera rotation and translation using essential matrix in epipolar geometry. The essential matrix is a 3×3 matrix that encodes the relationship between two images of the same scene from different viewpoints. The essential matrix is defined as in Eq. 4.

$$E := R \times [\boldsymbol{t}]_x \in \mathbb{R}^{3\times 3} \quad (4)$$

Where $[\boldsymbol{t}]_x$ is the matrix corresponding to cross product of $\boldsymbol{t}$ and R, is rotation matrix. As a property of the essential matrix for each P and P' Eq. 5 is held

$$(P')^T E\, P = 0 \quad (5)$$

As a result of essential matrix properties there are solutions for determining R and $\boldsymbol{t}$ based on performing Singular Value Decomposition. Using Multiple View Geometry [33] the rotation matrix R and cross product vector as shown in Eq. 6 and 7 are computed next

$$[\boldsymbol{t}]_x = \pm \mathrm{V\,W\,\Sigma\,V^T} \tag{6}$$

$$\mathrm{R} = \mathrm{U\,W^{-1}}V^T \tag{7}$$

$$\text{Where,}\quad \mathrm{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{8}$$

Where U and V are orthogonal 3×3 matrices and Σ is 3×3 diagonal matrix, see Eq. 9.

$$\Sigma = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{9}$$

There are several algorithms such as five-point and eight-point algorithms for solving R and $\boldsymbol{t}$. Five-point algorithm [34] was used in this study to estimate camera-rotation and translation from two consecutive images.

3.3 Most Reliable Straight Line

Let C_a and C_b be two arbitrary captured images. By assumption there is no moving object in the images if three landmarks exists such that the coordinates $P_a = P_b'$ where the position of the camera in those moments was identical. Using this fact, if the sums of all RSLs length converge to zero over time, it can be concluded that the camera is stabilized. Therefore the goal is to minimize the RSLs by issuing relevant commands to Quadcopter. Length of the RSL is directly correlated with accuracy of algorithm. When one RSL is longer than the others, stabilization is more accurate and slope is good measure for reliability of a specific RSL. When the slopes are not the same, which means either an object is moving inside the image or camera is rotating around itself.

The MRSL is the longest of RSLs those with similar slope. For each MRSL, pp', there exist a vector named Most Reliable Straight Vector (MRSV) at starting point of p' and endpoint of p. MRSV has equal magnitude and direction that of MRSL.

Although MRSL can be implemented to stabilize Quadcopters, but to remove all the unwarranted disturbances an alternative RSL should be used in order to avoid the risk of failure. In this case the RSLs with the same slopes are grouped into 3 equal size clusters. Beside the MRSL alternative RSLs one line from first along with one from second and two from the last clusters are selected and represented by an array$[\boldsymbol{ID},\ (\boldsymbol{x},\boldsymbol{y}),\ (\boldsymbol{x}',\boldsymbol{y}')\ ,\boldsymbol{L}]$, where ID is unique identification number for each landmark, $(\boldsymbol{x},\boldsymbol{y})$ is coordinate pair of $\boldsymbol{C_1}$, and $(\boldsymbol{x}',\boldsymbol{y}')$ is coordinate pair of $\boldsymbol{C_2}$, and L is the length of RSL.

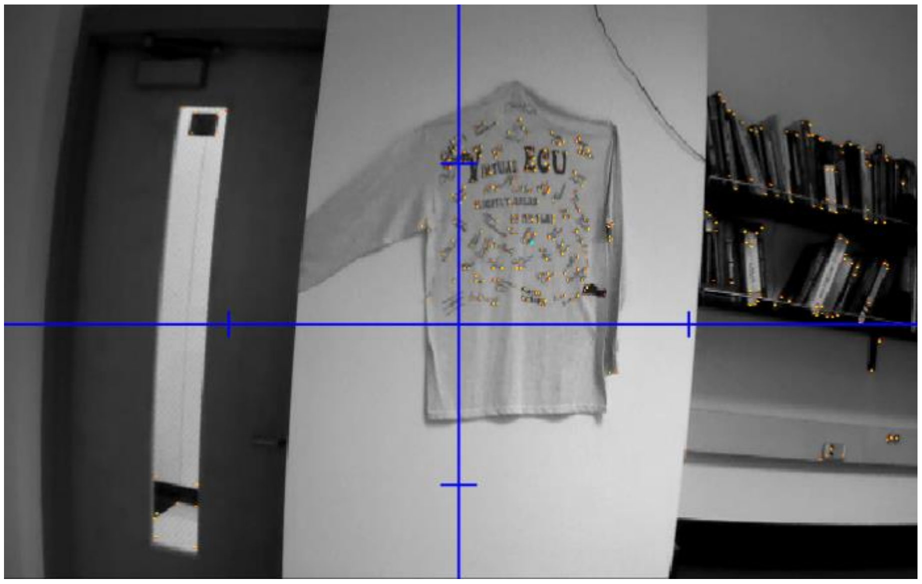

Figure 8: Landmark extraction initialization

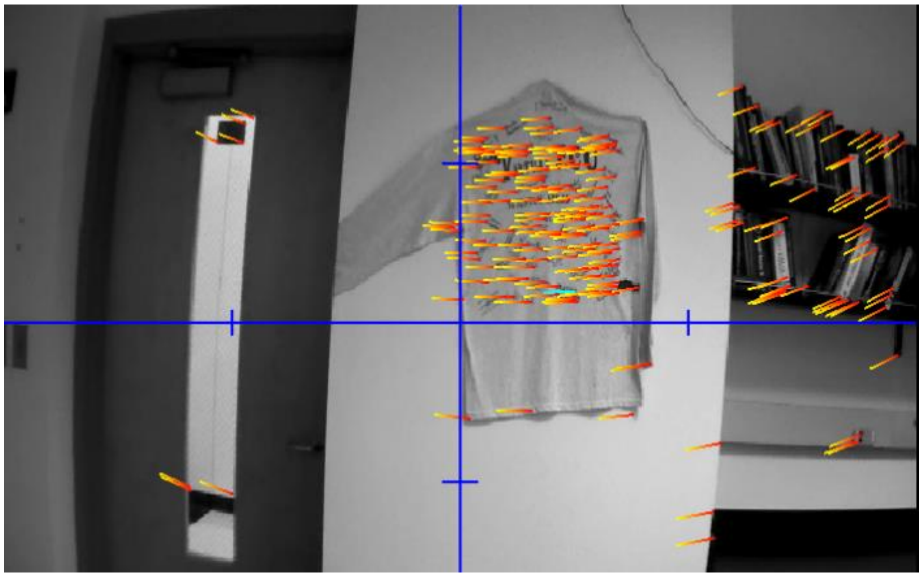
Figure 9 All detected RSLs

The candidate MRSL and 4 other RSL are used as inputs to the NN, Figure 8 shows the initial step and Figure 9 shows all detected RSLs.

3.4 Neural Network-Based Stabilization System

Before going through design and implementation, a brief introduction of Neural Network (NN) would be present. The simplest definition of a NN, more properly referred to as an 'artificial' neural network (ANN), is provided by the inventor of one of the first neurocomputers, Dr. Robert Hecht-Nielsen. He defines a neural network as: "...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs." [35]

ANNs are processing devices (algorithms or actual hardware) that are loosely modeled after the neuronal structure of the mammalian cerebral cortex but on much smaller scales. A large ANN might have hundreds or thousands of processor units, whereas a mammalian brain has billions of neurons with a corresponding increase in magnitude of their overall interaction and emergent behavior. Although ANN researchers are generally not concerned with whether their networks accurately resemble biological systems, some have. For example, researchers have accurately simulated the function of the retina and modeled the eye rather well [36].

Neural neworks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers' where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer' where the answer is output as shown in figure 10 [36]:

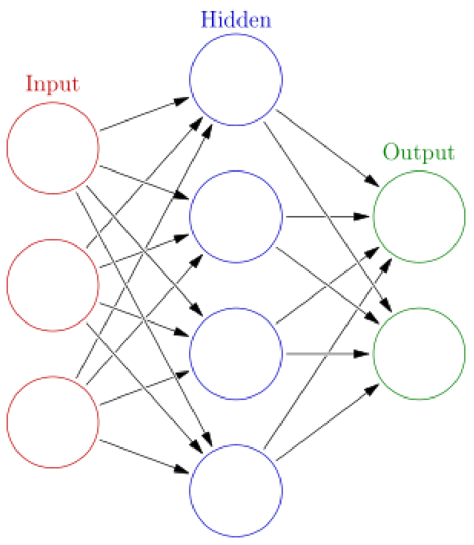

Figure 10 : Neural Network Schema

Most ANNs contain some form of 'learning rule' which modifies the weights of the connections according to the input patterns that it is presented with. In a sense, ANNs learn by example as do their biological counterparts; a child learns to recognize dogs from examples of dogs.

Although there are many different kinds of learning rules used by neural networks, this demonstration is concerned only with one; the delta rule. The delta rule is often utilized by the most common class of ANNs called 'backpropagational neural networks' (BPNNs). Backpropagation is an abbreviation for the backwards propagation of error [36].

With the delta rule, as with other types of backpropagation, 'learning' is a supervised process that occurs with each cycle or 'epoch' (i.e. each time the network is presented with a new input pattern) through a forward activation flow of outputs, and the backwards error propagation of weight adjustments. More simply, when a neural network is initially presented with a pattern it makes a random 'guess' as to what it might be. It then sees how far its answer was from the actual one and makes an appropriate adjustment to its connection weights[36].

For this thesis gradient descent used as an optimization method for backpropagation. Gradient descent is a first-order optimization algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point. The method calculates the gradient of a loss function with respect to all the weights in the network. The gradient is fed to the optimization method which in turn uses it to update the weights, in an attempt to minimize the loss function. The psuedocode of the algorithm is as below[37]:

```
initialize network weights (often small random values)
do
  forEach training example ex
    prediction = neural-net-output(network, ex)  // forward pass
    actual = teacher-output(ex)
    compute error (prediction - actual) at the output units
    compute $\Delta\omega_h$ for all weights from hidden layer to output layer  // backward pass
    compute$\Delta\omega_i$  for all weights from input layer to hidden layer   // backward pass continued
    update network weights // input layer not modified by error estimate
until all examples classified correctly or another stopping criterion satisfied
return the network
```

The goal of using NN is that by giving the $|MRSL| > 0$, suitable commands are issued as output. These commands counteract and move the Quadcoter back to its initial position. The designed NN has 7 inputs, where five of them are described in the previous section and are MRSL and 4 alternative RSLs. The built-in gyroscope data and speed of each motors are used as the remaining two inputs. Current speed of the motors helps in issuing commands with respect to current state of Quadcopter.

By controlling speed of each of the motors of a Quadcopter, its movement can be controlled in 3-D environment. The four different output of NN are respective values of electrical current applied to each brushless motor. The range of each output node is $[0,1]$. Zero means that the motors are off and the one means that the motors are running in the full load. As a last piece of design, NN has one hidden layer consisted of 10 nodes.

In order to train NN, a training dataset is generated that includes

$$(\boldsymbol{MRSL}, \boldsymbol{RSL_1}, \boldsymbol{RSL_2}, \boldsymbol{RSL_3}, \boldsymbol{RSL_4}, \boldsymbol{Gyro_t}, \boldsymbol{M_t}) \rightarrow (\boldsymbol{M_1}, \boldsymbol{M_2}, \boldsymbol{M_3}, \boldsymbol{M_4}) \qquad (10)$$

The 5 lines, gyroscope data and current state of motors, maps to 4-tuple electrical currents controlling the speed of each motor. The interpretation is that in order to move the camera to have a specific |MRSL|, a specific electrical current should be applied to each motor.

This dataset was generated in a creative way. A program generated 4 random numbers in the range $[0,1]$. These numbers were sent to motors as control commands. By capturing two images (C_0 and C_1) before and after sending that command, landmarks are extracted from C_0 and tracked in C_1. All variables regarding the MRSL algorithm were calculated and collected in a form of tuple as shown in (7). This experiment was repeated 700 times to make sure that enough data was generated for training. This experiment was supervised by an operator to avoid collisions and gather different viewpoints to make sure that the algorithm is not biased with a particular landmark. Noise and outliers were injected to the dataset to avoid over fitting. The final dataset contains 1000 samples, 90% were used for training and 10% for testing purpose.

Back propagation [38] was used to compute all the weights in the network. After training, the algorithm was validated using the test dataset. The success rate of 94.3% was observed.

CHAPTER 5 – THE COORDINATOR

Coordinator is an interface, which handles wireless communication between the Quadcopter and the PC. Because two different platforms are used in implementation, coordinator interprets the output of NN to suitable commands and transmits them to the Quadcopter's flight control board. Also it takes images from Quadcopter and sends them as input for MRSL algorithm. Moreover, coordinator is responsible for the emergency situation e.g. when the battery is too low or the MRSL algorithm is not responding for any reason, coordinator would do an emergency landing to reduce the probable damages. In this thesis Robot Operating System (ROS) was used. ROS is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools and it is open source. ROS provide a driver for Parrot AR.Drone and also running FAST algorithm on the captured images.

CHAPTER 6 – THE RESULTS

In this section the accuracy of stabilization and robustness of the Quadcopter using the MRSL approach is evaluated with experimental data obtained from several test flights in different location. The experiments can be categorized in two main groups, i) evaluating the stabilization without any external disturbances, b) applying disturbances e.g. pushing or pulling Quadcopter or blowing air with blower.

In the first set of experiments the Quadcopter was stabilized with supervision of a human intervention in a certain position, then the intervention stopped navigating allowing the algorithm to begin controlling Quadcopter for 60 seconds. The goal of the experiment was to evaluate the stability of the Quadcopter. This experiment was repeated 100 times. Two criteria measured for evaluation were movement of the detected landmarks and movement of the center of mass of the Quadcopter from viewpoint of external fixed camera. As is shown in Figure 11 visualization of the movement of the landmarks and the actual position of the Quadcopter in a 3-D space.

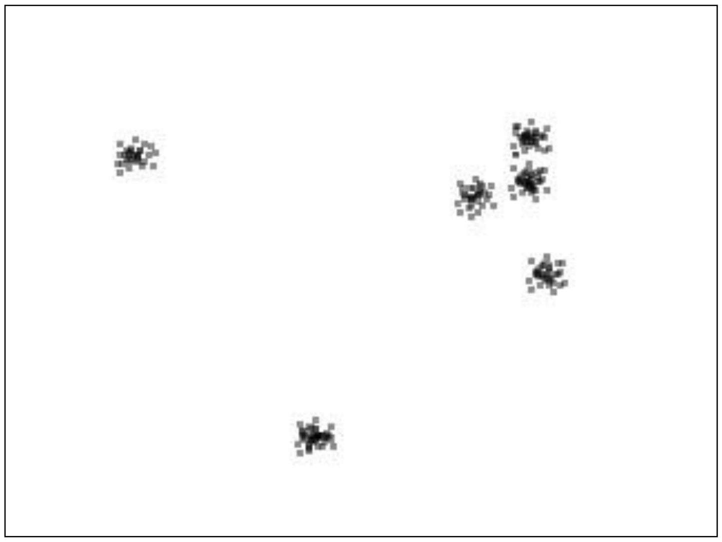

Figure 11 Landmarks change captured by camera

Figure 12 shows the movement of the center of mass of the Quadcopter captured by a fixed camera. Accuracy of stabilization measured is expressed by the shortest diameter of a circle that includes all the landmarks. In our experiment the average diameter was about 15 inches for center of mass of Quadcopter.

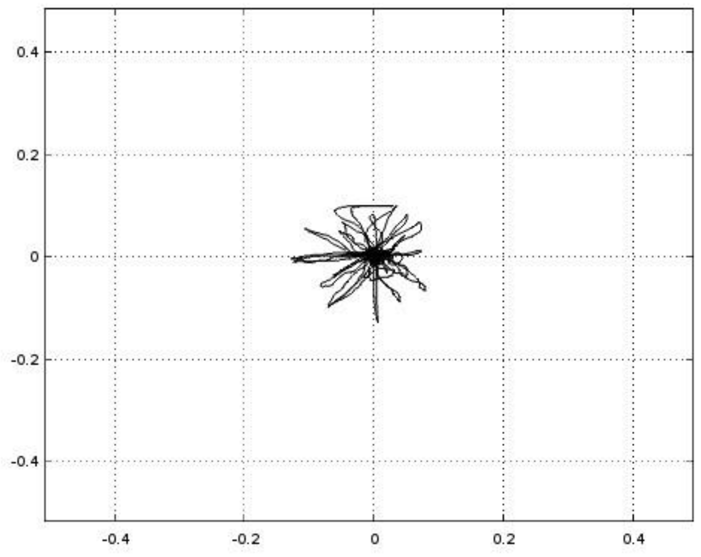

Figure 12 : Movement of center of Quadcopter captured by fix camera

In the second experiment after stabilization, external disturbances were introduced in order to test robustness of the MRSL algorithm. The goal was to measure convergence speed and time of the Quadcopter to return to its initial position. Result showed that regardless of the direction of the introduced forces to disturb the stability of the Quadcopter, the convergence time remained almost constant. But the magnitude of force is inversely correlated with convergence time. The algorithm was not robust to forces that change the direction of the camera where, none of the landmarks remained within the image. This threshold is the critical point of MRSL algorithm. As Figure 13 shows forces from 1 to 5 Newton were applied and the measure of the convergence time was observed.

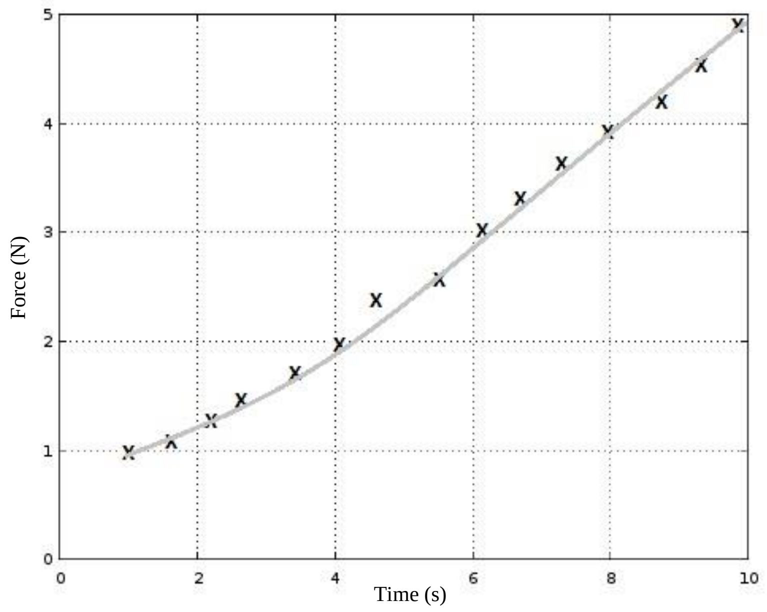

Figure 13 Convergence time by amount of applied force

CHAPTER 7 – CONCLUSION

Stabilizing and localizing positioning systems autonomously in the areas without GPS accessibility is a difficult task. In this paper, we introduce an innovative methodology for stabilizing and localizing Quadcopters in 3-D environments. Most of the current methods used to positioning objects in 3-D are using expensive equipment in contrast to the methodology introduced in this paper. To prove the robustness of the algorithm, an experiment was set up using a Quadcopter to measure the reliability of the algorithm, it was observed that the algorithm can stabilize the Quadcopter effectively even in the presence of external.

REFERENCES

[1] Javier Irizarry, Masoud Gheisari, Bruce N. Walker (2012) Usability assessment of drone technology as safety inspection tools, ITcon Vol. 17, pg. 194-212

[2] Staff writer (25 April 2010). "Robot subs trying to stop Gulf oil leak". CBC News. Retrieved 25 April 2010

[3] Brando, A (2003) Firefighter-robot interaction during a hazardous materials incident exercise, 2011 15th International Conference on Advanced Robotics

[4] Jakob Julian Engel (2011), Autonomous Camera-Based Navigation of a Quadrocopter, TUM University Thesis

[5] William Morris, Ivan Dryanovski, Jizhong Xiao (2010) 3D indoor mapping for micro-UAVs using hybrid range finders and multi-volume occupancy grids, workshop on RGB-D: Advanced Reasoning with Depth Cameras, Zaragoza, Spain

[6] Erdinç Altu˘g , James P. Ostrowski , Camillo J. Taylor (2005) Control of a Quadrotor Helicopter Using Dual Camera Visual Feedback, The International Journal of Robotics Research

[7] Siciliano B, Khatib O (2008) Handbook of Robotics, Springer

[8] C. Stachniss. (2006) Exploration and Mapping with Mobile Robots. PhD thesis, University Freiburg

[9] Leonard, Durrant-Whyte: Mobile robot localization by tracking geometric beacons:

[10] Smith, Self, Cheesman: Estimating uncertain spatial relationships in robotics

[11] Riisgaard, Søren and Blas, Morten Rufus. SLAM for Dummies: A Tutorial Approach to Simultaneous Localization and Mapping . Boston: MIT, 2004.

[12] C. Stachniss. Exploration and Mapping with Mobile Robots. PhD thesis, Universi¨at

Freiburg, 2006.

[13] Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM: A Factored Solution to

Simultaneous Localization and Mapping Problem. In: National Conference on Artificial

Intelligence (2002)

[14] Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM 2.0: An Improved

Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably

Converges. In: International Joint Conference on Artificial Intelligence (2003)

[15] Davison, A., Reid, I., Molton, N.D., Stasse, O.: MonoSLAM: Real-time single camera SLAM. IEEE Transaction on Pattern Analysis and Machine Intelligence 29(6), 1052–1067 (2007)

[16] Klein, G., Murray, D.: Parallel Tracking and Mapping for Small AR Workspaces. In: 6th IEEE International Symposium on Mixed and Augmented Reality, pp. 225–234 (2007)

[17] Yoo, Jaesang, Kyusung Cho, Jinki Jung, and Hyun S. Yang. "Online scene modeling for interactive AR applications." In Entertainment Computing-ICEC 2010, pp. 139-150. Springer Berlin Heidelberg, 2010.

[18] Nevado, M.M., Gercía-Bermejo, J.G., Zalama, E.: Obtaining 3D models of indoor environments with a mobile robot by estimating local surface directions. Robotics and Autonomous Systems 48(2-3), 131–143 (2004)

[19] Viejo, D., Cazorla, M.: 3D plane-based egomotion for SLAM on semi-structured environment. In: IEEE International Conference on Intelligent Robots and Systems (2007)

[20] Rachmielowski, A., Jägers, M., Cobzas, D.: Realtime visualization of monocular data for 3D reconstruction. In: Canadian Conference on Computer and Robot Vision, pp. 196–202 (2008)

[21] Chekhlov, D., Gee, A., Calway, A., Mayol-Cuevas, W.: Ninja on a plane: Automatic Discovery of Physical Planes for Augmented Reality Using Visual Slam. In: 6th IEEE International Symposium on Mixed and Augmented Reality, pp. 1–4 (2007)

[22]Gee, A., Chekhlov, D., Calway, A., Mayol-Cuevas, W.: Discovering Higher Level Structure in Visual SLAM. IEEE Trans
actions on Robotics, 980–990 (2008)

[23] Wikipedia. Quadrotor — Wikipedia, the free encyclopedia, 2011. [http://en. wikipedia.org/w/index.php?title=Quadrotor&oldid=443167665].

[24] Hoffmann, G.M.; Rajnarayan, D.G.; Waslander, S.L.; Dostal, D.; Jang, J.S.; Tomlin, C.J. (November 2004). "The Stanford Testbed of Autonomous Rotorcraft for Multi Agent Control (STARMAC)" (PDF). In the Proceedings of the 23rd Digital Avionics System Conference. Salt Lake City, UT. pp. 12.E.4/1–10.

[25] Stafford, Jesse (Spring 2014). "How a Quadcopter works | Clay Allen". University of Alaska, Fairbanks. Retrieved 2015-01-20.

[26] Stafford, Jesse (2014-01-12). "How does a Quadcopter fly | Minicopter-jp.com". Minicopter-jp.com. Retrieved 2015-01-20.

[27] Product specification guide at http://ardrone2.parrot.com/

[28] Parrot. AR-Drone developer guide for SDK 1.6, 2011. [http://projects.ardrone.org].

[29] D. Lowe. Object recognition from local scale-invariant features. In Proc. of the International Conference on Computer Vision (ICCV), 1999.

[30] H. Bay, T. Tuytelaars, and L.V. Gool. SURF: Speeded-up robust features. In Proc. of the European Conference on Computer Vision (ECCV), 2008.

[31] E. Rosten and T. Drummond (2006) Machine learning for high-speed corner detection. In Proc. of the European Conference on Computer Vision (ECCV)

[32] Harris, C., & Stephens, M. (1988, August). A combined corner and edge detector. In Alvey vision conference (Vol. 15, p. 50).

[33] Zisserman, R. H. (2004). Multiple View Geometry in Computer Vision. Cambridge University Press.

[34] H. Li and R. Hartley. Five-point motion estimation made easy. In Proc. of the International Conference on Pattern Recognition (ICPR), 2006

[35] "Neural Network Primer: Part I" by Maureen Caudill, AI Expert, Feb. 1989

[36] What is nural network , Josef Burger, 2010

[37] Gradient descent , en.wikipedia.org/wiki/Gradient_descent

[38] Yves Chauvin (2008) Back-Propagation: Theory, Architecture, and Applications, Taylor & Francis