GROUND PLANE DETECTION USING AN RGB-D CAMERA FOR QUADCOPTER LANDING

DOĞAN KIRCALI
B.S., Electronics Engineering, Işık University, 2009

Submitted to the Graduate School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science
in
Electronics Engineering

IŞIK UNIVERSITY
2013

IŞIK UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

GROUND PLANE DETECTION USING AN RGB-D CAMERA FOR QUADCOPTER LANDING

DOĞAN KIRCALI

APPROVED BY:

Assist. Prof. F. Boray Tek (Thesis Supervisor)	Işık University	____________________
Assoc. Prof. Hasan F. Ateş	Işık University	____________________
Assist. Prof. Emine Ekin	Işık University	____________________

APPROVAL DATE: ..../..../....

GROUND PLANE DETECTION USING AN RGB-D CAMERA FOR QUADCOPTER LANDING

Abstract

The purpose of this study is to build an autonomous quadcopter which is capable of automated detection of landing zones and landing. To ensure this there are important steps linked together.

The first step is to build a quadcopter that can fly. We have used a commercially available quadcopter platform and an Ardu Pilot Mega control card to build a low cost, easy to implement, and stable platform. An open source firmware, named Arducopter, is used for the control card. This system can take-off, land, hover, and follow a given flight path.

In order to detect of landing zones and safe landing, we propose to use an RGB-D camera as a sensor and a small onboard pc as the computing engine. Hence, we have modified the acquired quadcopter frame to integrate additional components. In this thesis, we propose a novel and robust ground plane and obstacle detection algorithm based on depth information using RGB-D camera. Moreover, our method was compared with V-disparity algorithm from the literature. It has been shown that our algorithm performs better than V-disparity method and produces useful ground plane-obstacle segmentations, even for difficult cases. The method is able to work in highly dynamic platforms. This algorithm is generic in the sense that it can be used for different forward-facing RGB-D placements, for example in ground vehicles or robots.

Moreover, we developed a pre-process to allow the use of the method for down-facing sensor view positions as the core method is inadequate for landing zone detection. The proposed method compensates the movements of the camera caused by the air vehicle, and detects the ground plane obstacles successfully. It has been shown that the use of RGB-D camera allows ground plane and landing zone detection even in no-light conditions.

All necessary components in this thesis were financed by FMV Işık University internal research funds BAP-10B302 project.

QUADCOPTER İNİŞİ İÇİN RGB-D KAMERASI İLE ZEMİN DÜZLEMİ ALGILAMA

Özet

Bu çalışmadaki amaç bir dört rotorlu ufak hava aracına özerk bir şekilde güvenli iniş alanı belirleme ve iniş yapma özelliği kazandırmaktır. Bunu sağlamak için birbirine bağlı önemli aşamalar vardır. Bu aşamaların ilki kararlı bir şekilde uçabilen dört rotorlu bir aracın kurulmasıdır.

Bu amaçla, düşük maliyetli, kolay kurulabilir, ve kararlı bir platform sunduğundan, ticari olarak mevcut olan ArduCopter ve Ardu Pilot Mega kontrol kartından oluşan platform ve açık kaynaklı ArduCopter yerleşik yazılımı kullanılmıştır. Bu haliyle sistem komutlar verilerek kalkış, iniş, asılı kalma ve uçuş rotasını takip etme fonksiyonlarını gerçekleştirebilmektedir.

Bu çalışmada güvenli inişe uygun iniş sahalarının belirlenmesi için RGB-Derinlik kamerası kullanılarak bir yöntem geliştirilmiştir. Bu nedenle, dört rotorlu hava aracı platformu değiştirilerek RGB-Derinlik kamerası ve bir güverte bilgisayarı entegre edilmiştir.

Bu çalışmada, Kinect derinlik kamerasından elde edilen derinlik bilgisine dayalı yeni ve gürbüz bir zemin düzlemi algılama algoritması önerilmektedir. Birçok farklı veri seti ile yöntemin başarılı bir şekilde çalıştığı gözlenmiştir. Yapılan karşılaştırmalı deneyler, önerilen algoritmanın mevcut V-disparity yönteminden daha iyi çalıştığını göstermektedir. Önerilen algoritma oldukça devinimli ortamlarda çalışabilmektedir. Ayrıca genele uygulanabilir bir yöntemdir.

Önerilen zemin düzlemi algılama yönteminin aşağı-dönük bir platform için uygun olmaması dolayısıyla; güvenli iniş alanı belirlemekte yetersiz olmasından dolayı; mevcut yöntemin kullanılmasına olanak sağlayacak bir ön işlem tanımlanmıştır. Önerilen yöntem, hava aracının hareketlerinden kaynaklı olan kamera görüş açısındaki değişimleri başarılı bir şekilde telafi edebilmekte ve zemin düzlemi üzerindeki nesneleri algılayabilmektedir. Önerilen yöntemin ışık olmayan koşullarda dahi başarılı bir biçimde çalıştığı yapılan deneylerle gösterilmiştir.

Bu projedeki ekipmanların tümü FMV Işık Üniversitesi iç araştırma BAP-10B302 projesi fonu ile finanse edilmiştir.

Acknowledgements

First of all, I would like to thank my supervisor Assist. Prof. F. Boray Tek for his advices, fatherly guidance and patience. During this study when I lost my concentration he was always there to keep me on the track. I also would like to thank to him for the recommendations to improve my quality of life. I consider it a great privilege working with him.

I would like to express my sincere thanks to my former teammate İbrahim K. İyidir who is also a valuable friend to me, for his advices and endless support.

Many thanks to a special person, Sezin Ata for her moral support and encouragement.

Last but certainly not least, I am very grateful to my family. I thank to my parents, Azize and Hüseyin for their patience and understanding.

To my family…

Table of Contents

Abstract ii

Özet iii

Acknowledgements iv

List of Tables viii

List of Figures ix

List of Symbols xi

List of Abbreviations xii

1 Introduction 1

2 Quadcopter 5
- 2.1 Hardware System 6
  - 2.1.1 Multirotor Frame 7
  - 2.1.2 Electronics 9
    - 2.1.2.1 Main Components 10
    - 2.1.2.2 Secondary Components 13
    - 2.1.2.3 Componenets for Autonomous Landing 14
    - 2.1.2.4 Useful Payload 16
  - 2.1.3 Overall System 17
- 2.2 Software System 20
  - 2.2.1 Onboard Software 20
  - 2.2.2 Ground Control Station 21
  - 2.2.3 MAVLink 22
  - 2.2.4 RAVLAB Control Station 22

3 Ground Plane Detection Using an RGB-D Sensor 26
- 3.1 Ground Plane Detection 27
  - 3.1.1 Detection for fixed pitch 28
  - 3.1.2 Detection for changing pitch and roll 29
- 3.2 Experiments 34

3.3 Discussions 42

4 Landing zone detection **43**

4.1 Method 45
4.2 Experiments 47
4.3 Discussions 49

Conclusion **61**

References **64**

Curriculum Vitae **71**

List of Tables

2.1 LiPo specifications 12
2.2 Complete system weight 19

List of Figures

1.1 Examples to the different categories of UAVs 2

2.1 Brief system layout 6
2.2 Multirotor configuration examples 8
2.3 Quadcopter frames 9
2.4 Quadcopter flight principles 10
2.5 LiPo charger, storage bag, damaged LiPo 11
2.6 RC Transmitter & receiver 12
2.7 Ardu pilot mega cards 13
2.8 Secondary components 14
2.9 FitPC2 15
2.10 Camera and downlink system 17
2.11 Gimbal system and 1/3″ CCD camera 18
2.12 Complete system layout 18
2.13 Our quadcopters 19
2.14 Flow chart of roll control 21
2.15 APM planner screens 23
2.16 RAVLAB Control Station screenshot 25

3.1 Axes & effect of the projection 29
3.2 Example depth map & effect of the pitch change 31
3.3 Exclusion rule explanation 33
3.4 Mobile robot platform 34
3.5 ROC curves & accuracy for data set 1 37
3.6 ROC curves & accuracy for data set 2 38
3.7 Experimental results for ground plane detection from different scenes 40
3.8 Experimental results for ground plane detection-cont. 41

4.1 Viewing perception & axes 46
4.2 Depth envelope for negative pitch angle 47
4.3 Experimental results for landing zone detection-1 50
4.4 Experimental results for landing zone detection-2 51
4.5 Experimental results for landing zone detection-3 52
4.6 Experimental results for landing zone detection-4 53
4.7 Experimental results for landing zone detection-5 54
4.8 Experimental results for landing zone detection-6 55

4.9 Experimental results for landing zone detection-7 56
4.10 Experimental results for landing zone detection-8 57
4.11 Experimental results for landing zone detection-9 58
4.12 Experimental results for landing zone detection-10 59

List of Symbols

y	Vertical position
x	Horizontal position
f	Pixel's depth value
T	Curve comparison threshold
D	Depth image
C_R	Reference ground plane curve
C_U	Column of depth map
h_r	Histogram of rth row
R	Number of rows
C	Number of columns
θ	Rotation angle
r	Row number
D_θ	Rotated depth image
E	Depth envelope
$A1$	Fixed method
$A2$	Pitch compensated method
$A3$	Pitch and roll compensated method

List of Abbreviations

APM	**A**rdu **P**ilot **M**ega
ESC	**E**lectronic **S**peed **C**ontroller
GCS	**G**round **C**ontrol **S**tation
GPS	**G**lobal **P**ositioning **S**ystem
GUI	**G**raphical **U**ser **I**nterface
ILS	**I**nstrument **L**anding **S**ystem
IMU	**I**nertial **M**easurement **U**nit
LADAR	**LA**zer **D**etection **A**nd **R**anging
LED	**L**ight **E**mitting **D**iode
LIDAR	**LI**ght **D**etection **A**nd **R**anging
LiFePo4	Lithium Iron Phosphate
LiPo	**Li**thium **Po**lymer
MAV	**M**icro **A**erial **V**ehicle
MAVLink	Micro Air Vehicle Communication Protocol
NAV	**N**ano **A**erial **V**ehicle
PID	**P**roportional **I**ntegral **D**erivative
PWM	**P**ulse **W**idth **M**odulation
RC	**R**adio **C**ontrol
RGB	**R**ed **G**reen **B**lue
RGB-D	**R**ed **G**reen **B**lue **D**epth
ROC	**R**eceiver **O**perating **C**urves
ROS	**R**obot **O**perating **S**ystem
RPV	**R**emotely **P**iloted **V**ehicle
SIFT	**S**cale **I**nvariant **F**eature **T**ransform

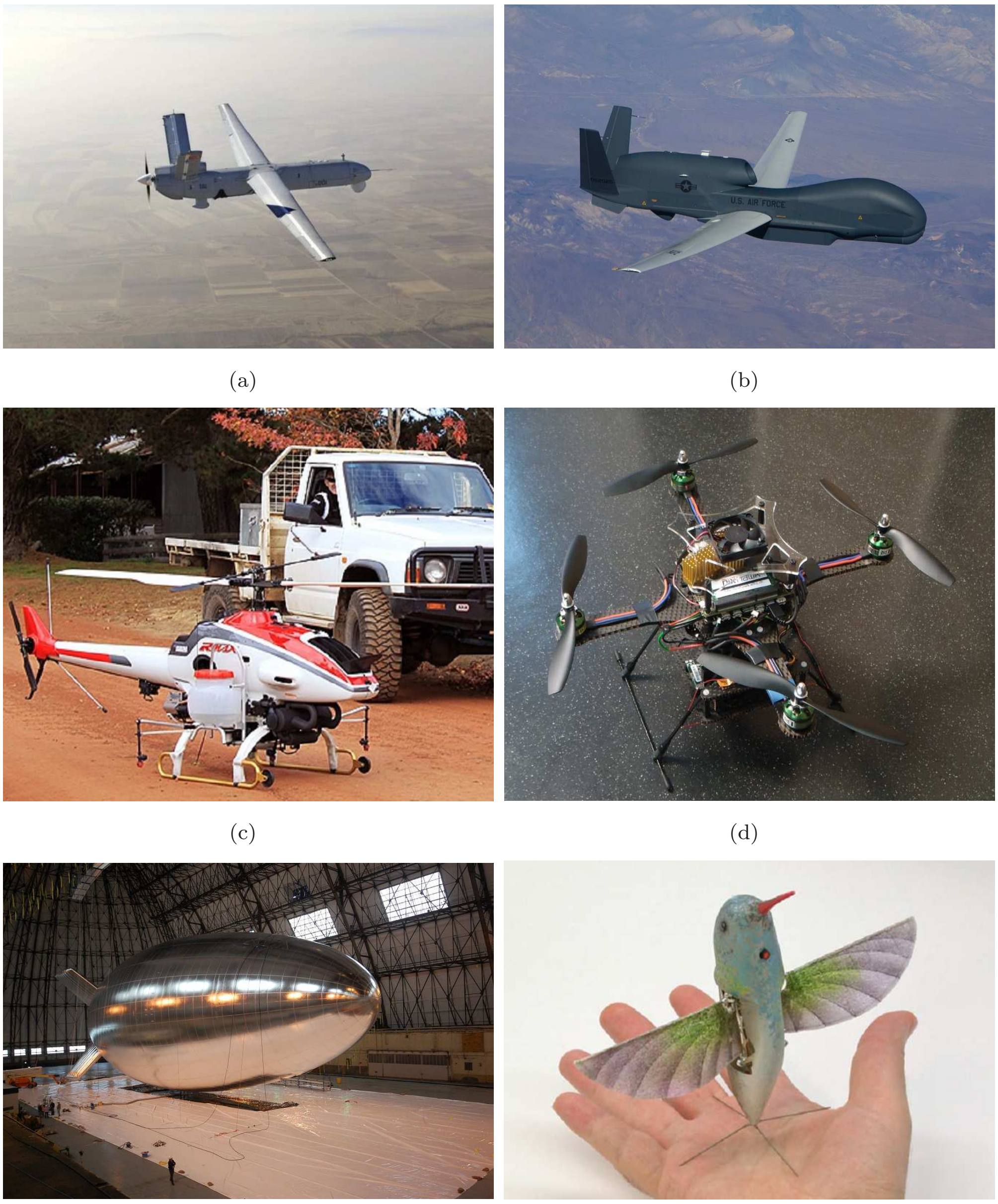
SLAD	**S**afe **L**anding **A**rea **D**etermination
SURF	**S**peeded **U**p **R**obust **F**eatures
TLS	**T**ransponder **L**anding **S**ystem
U-SURF	**U**pright **S**peeded **U**p **R**obust **F**eatures
UAV	**U**nmanned **A**erial **V**ehicle
USB	**U**niversal **S**erial **B**us
VTOL	**V**ertical **T**ake **O**ff and **L**anding
PC	**P**ersonal **C**omputer
LMedS	**L**east **Med**ian **S**quares

Chapter 1

Introduction

Development of unmanned air vehicles (UAVs) has been rapidly increasing in the recent years. UAVs offer advantages when used for aerial surveillance, reconnaissance, and inspection in complex and dangerous environments. Indeed, UAVs are preferred for dull, dirty, or dangerous missions because of their expense compared to a full scale manned vehicle. Research and production of UAVs, especially micro aerial vehicles (MAVs), usually named as drones, are popular due to cheaper and more robust electronics. UAVs are being used mostly by military but they are also used in a small but growing number of civil applications because of the infinite possibilities of utilizing their characteristics for civil applications. Some examples of these applications are firefighting, surveillance, aerial photography, mapping, and agricultural-chemical spraying.

UAV platforms can be classified according to their flight characteristics in four categories: fixed wings, rotorcrafts, blimps and flapping wings. Fixed wing UAVs require a runway to take-off and land, or catapult launching. As the vertical lift is achieved from the wings, they can have long flight times and can fly at high cruising speeds (Figure 1.1(a)-(b)). Rotorcrafts are vertical take-off and landing (VTOL) capable vehicles that have the advantages of hovering capability and high maneuverability (Figure 1.1(c)-(d)). However as the lift is produced by the rotors, they are more power hungry than any other system. Blimps are lighter than air so they float in the air (Figure 1.1(e)). They are generally large sized,



(a) (b) (c) (d) (e) (f)

Figure 1.1: Examples of different UAVs: (a) TAI Anka [1]; (b) Northrop Grumman RQ-4 Global Hawk [2]; (c) Yamaha Rmax [3]; (d) ETH Zurich PIXHAWK [4]; (e) Lockheed Martin High Altitude Airship [5]; (f) AeroVironment NAV [6]

slow flying vehicles with high endurance. Flapping wing UAVs also called as ornithopters (Figure 1.1(f)), have wings inspired from birds and flying insects, also they are usually nano aerial vehicles (NAVs).

There are also some hybrid configurations which can take-off vertically as rotorcraft than transform into a fixed wing aircraft by tilting their rotors.

UAVs come in two varieties: some are controlled from a remote location, called as remotely piloted vehicles (RPVs), and others fly autonomously based on pre-programmed flight plans using more complex dynamic automation systems. Remote controlled ones are becoming less preferred as they still need humans and hours of training is required to use them. On the other hand, autonomous flight requires complex algorithms and expensive hardware. Under the autonomous controller, the tasks can be categorized as: take-off, flight planning, landing zone detection and landing, while flight planning can be straight forward or a more complex depending on the environment.

Because of the physical complexity of a rotorcraft vehicle and required complex movement algorithms, fixed wing UAVs like USAF's MQ-1 Predator[7] and RQ-4 Global Hawk[2], are preferred. As they need to fly horizontally to stay in the air, their algorithms are simply going in the direction where there are no obstacles. Helicopter UAVs like Austrian Schiebel Camcopter S-100[8] and United States' Northrop Grumman MQ-8 Fire Scout[9] become available in 2003 and both of them are not fully autonomous when they were introduced, fully autonomous flight with a full scale helicopter achieved in 2004 with Boeing Little Bird[10].

The navigational control decides the route to the GPS point that is given by the user. The main concern is avoiding the obstacles while maintaining a continuous flight. Ground plane detection and obstacle detection are essential tasks to determine passable regions for autonomous navigation. To detect the ground plane in a scene the most common approach is to utilize depth information (i.e. depth map). Various methods and sensors have been used to compute the depth map of the scene. The methods can be categorized as active or passive [11]. Active methods use additional sensors such as 3D lasers, ultrasonic range finders or IR projectors. Passive methods consist of optical flow and stereo vision algorithms where the computation is more emphasized.

Moreover, many researches are focused on complex autonomous flight planning as in [12, 13]. At the end of the flight planning task, after the final destination point is reached it is important to find a proper place for landing. For larger aircrafts landing zone is usually a runway in an airfield, where already available systems like ILS or TLS can be used. If similar systems are not present, vision based methods can be used as in [14]. Smaller aircrafts also requires a runway but a road or an empty field can be used as landing zone. However VTOL vehicles like helicopters, require an empty space slightly larger than the vehicle itself as a landing zone, which is the main advantage of the VTOL vehicles. So the detection of landing zone for VTOL vehicles is more complex than the detection for the fixed-wing aircrafts. Advances in unmanned rotorcraft systems are studied in [15].

In this thesis, we aim to build an autonomous quadcopter which is capable of automated landing. Our autonomous quadcopter flight software is based on a popular framework "ArduCopter" [16], which is capable of: take-off, landing, hovering, and point to point navigation *on command*. While we aim to automate all of these tasks, we are mainly focused on automated landing .

In Chapter 2, we discussed the hardware and software components of our quadcopter platform. Every component and the key aspects of selecting those components are investigated.

In Chapter 3, a novel ground plane detection algorithm using an RGB-D camera is introduced for mobile robot navigation. Introduced method is compared with an existing method from the literature, i.e. V-disparity algorithm [17].

In Chapter 4, we adapted the method that we describe in Chapter 3 to detect possible safe landing zones. Thus, we propose a new improved method to detect landing using an RGB-D camera.

Chapter 2

Quadcopter

Interest for multirotor aircrafts are increasing rapidly due to smaller and lower cost microprocessors and sensors. Individuals are building small multirotor platforms and flying them with RC remotes as a hobby (Remotely Piloted Vehicles). Meanwhile academic and research communities around the world are working on the development of autonomous multirotors for specialized missions.

This thesis is a part of a project in which we aim to build an autonomous multirotor platform, a.k.a. quadcopter. Our autonomous quadcopter system consists of a base quadcopter frame, motor control electronics, a control card, sensors for flight control, a small size compact PC, an RGB-D (Red Green Blue-Depth) camera, a sonar sensor and useful payload such as a camera. A brief layout of our system can be seen in Figure 2.1. The factors which affected our choice for the quadcopter platform and our modifications on it are explained in Section 2.1.1. In Section 2.1 hardware components of the system, in Section 2.2 software systems are explained in more detail. Electronic components in the quadcopter including necessary, optional components and as useful payload are discussed in Section 2.1.2, completed platform is discussed in Section 2.1.3.

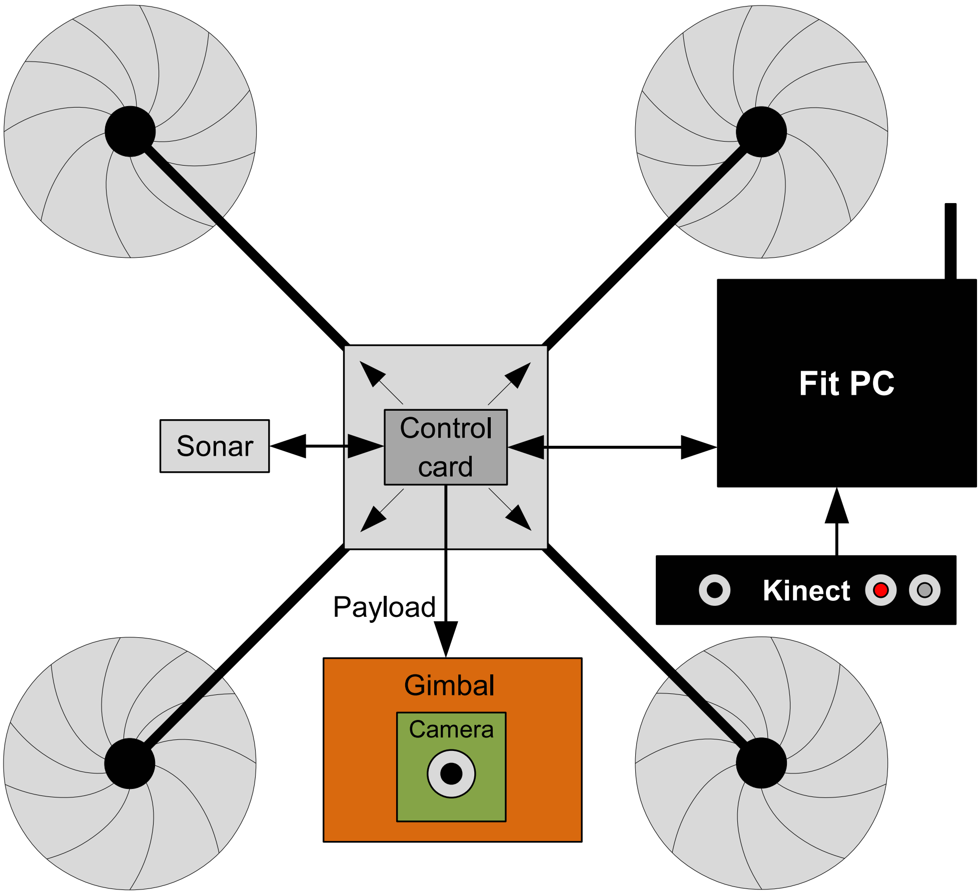


Figure 2.1: Brief system layout

2.1 Hardware System

Multirotors use two or more propellers to obtain the lift required to fly. Conventional helicopters are not accepted as multirotors since the tail rotor does not provide lift. The forces that apply on a multirotor platform should be dynamically controlled to obtain a stable flight. In this way crashes are avoided and hovering is possible. Calculating the position and the movements of the platform and controlling the variations of each rotor's lift, also known as the attitude control [18, 19], is essential. Control cards with an inertial measurement unit (IMU) are used to overcome this problem.

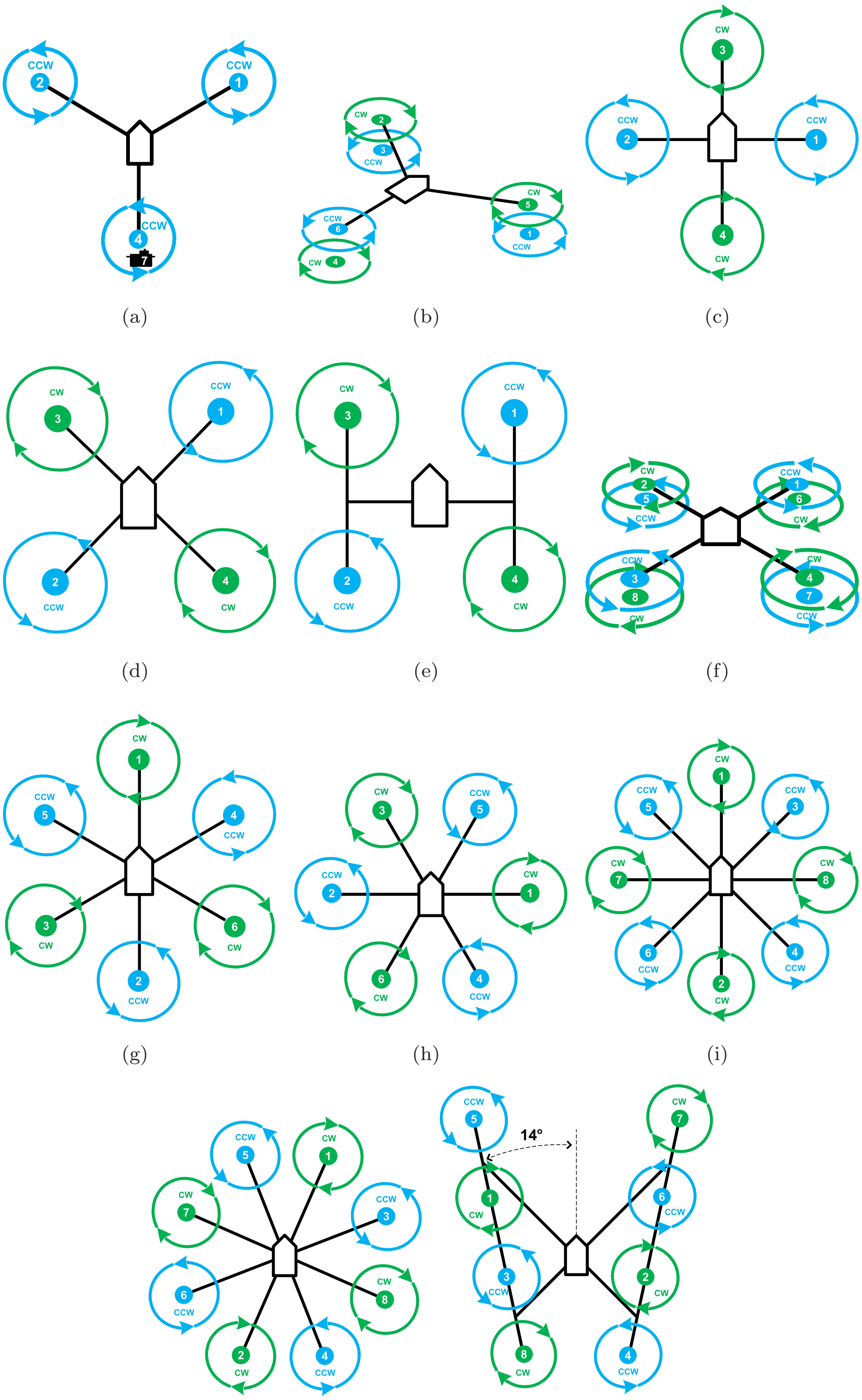
All necessary components in this thesis were financed by FMV Işık University internal research funds BAP-10B302 project.

2.1.1 Multirotor Frame

Multirotor platforms can be built in any shape desired with different number of rotors as long as the motor locations are known. Common multirotor configurations are shown in Figure 2.2. It can be seen that a single frame can be used for different configurations by changing the orientation of the control card. Adding more than four rotors does not show much improvement for stability, however one can gain more lifting capacity and fault tolerance with more motors. We have chosen to use quadcopter configuration which provided high stability with least number of motors, in addition to being cost effective.

There are many manufacturers that offer multirotor frames. We used Jdrones Arducopter Quad Frame V1(Figure 2.3(a)) and HobbyKing Quadcopter Frame V1(Figure 2.3(b)). Jdrones frame is 62cm wide (rotor to rotor) and has aluminum rotor arms whereas HobbyKing frame is 55cm wide and completely made of plywood. Although they use the same equipment these two frames exhibit different flight characteristics because of the width of the frame and the construction material. Wider rotor arms give more stability but slower response to inputs. Aluminum rotor arms conduct any vibration caused by the rotating propellers whereas plywood absorbs most of the vibration. Any vibration that is transmitted to the center of the frame, where all the electronics are placed, reduce the stability by causing sensor measurement errors.

Quadcopter X configuration's flight principles are shown in Figure 2.4. Lift is obtained from the rotors' total force. In order to move forward, quad has to pitch by rear rotors speed up while front rotors slow down (Figure 2.4(a)). To move to the sides rotors in the desired direction slow down and the ones in the opposite side speed up as shown in Figure 2.4(b). Anti rotational force is the key factor for yaw control, to turn clockwise rotors that are turning counter-clockwise speed up and the ones which spin clockwise slow down, as the anti-rotational force is no longer in equilibrium, the quad yaws (Figure 2.4(c), [18]).



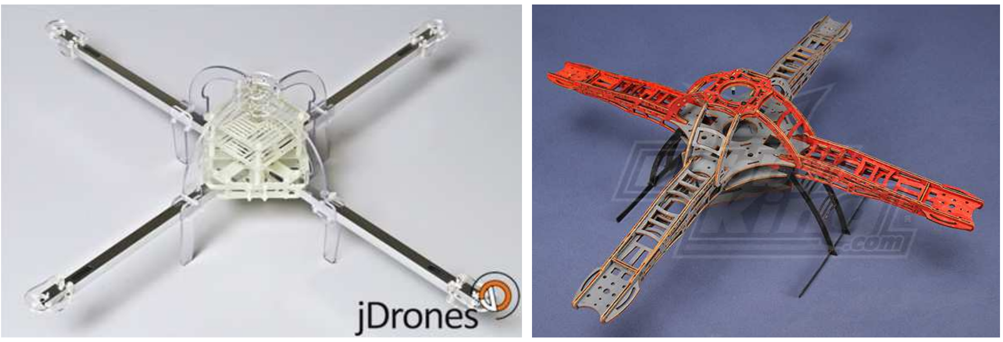
(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k)

Figure 2.2: Multirotor configuration examples: (a) tricopter; (b) hexa tricopter (Y6); (c) quadcopter plus configuration; (d) quadcopter X configuration; (e) quadcopter H configuration; (f) octa quadcopter; (g) hexacopter plus configuration; (h) hexacopter X configuration; (i) octacopter plus configuration; (j) octacopter X configuration; (k) octacopter V configuration



(a) (b)

Figure 2.3: Quadcopter frames: (a) Jdrones Arducopter quad frame V1[20]; (b) HobbyKing quadcopter frame V1[21]

We mostly use Jdrones frame because of its rigidity and slower response and easy to modify. However we made some changes to the frame for our needs. First of all ground clearance of the original frame was very low and the supplied plexiglass landing skids and dome was very fragile. Thus we removed the dome and landing skids and installed 130mm carbon fiber landing gears for planes. After earning sufficient ground clearance we added extra brackets made of PVC foam board, for on board FitPC computer and our payloads. Also we have built demountable collision protection bumpers made of Styrofoam to reduce damages from possible bad landing and crashes.

Because of the simplicity we first set the quadcopter in the plus configuration but then we used the X configuration to give a forward-looking camera an unobstructed field of view (Figure 2.2(d) vs Figure 2.2(c)).

2.1.2 Electronics

Electronics that are used in the quadcopter can be categorized as main components such as control card, electronic speed controllers (ESCs), secondary components like battery monitor, telemetry, and useful payload like camera and gimbal.

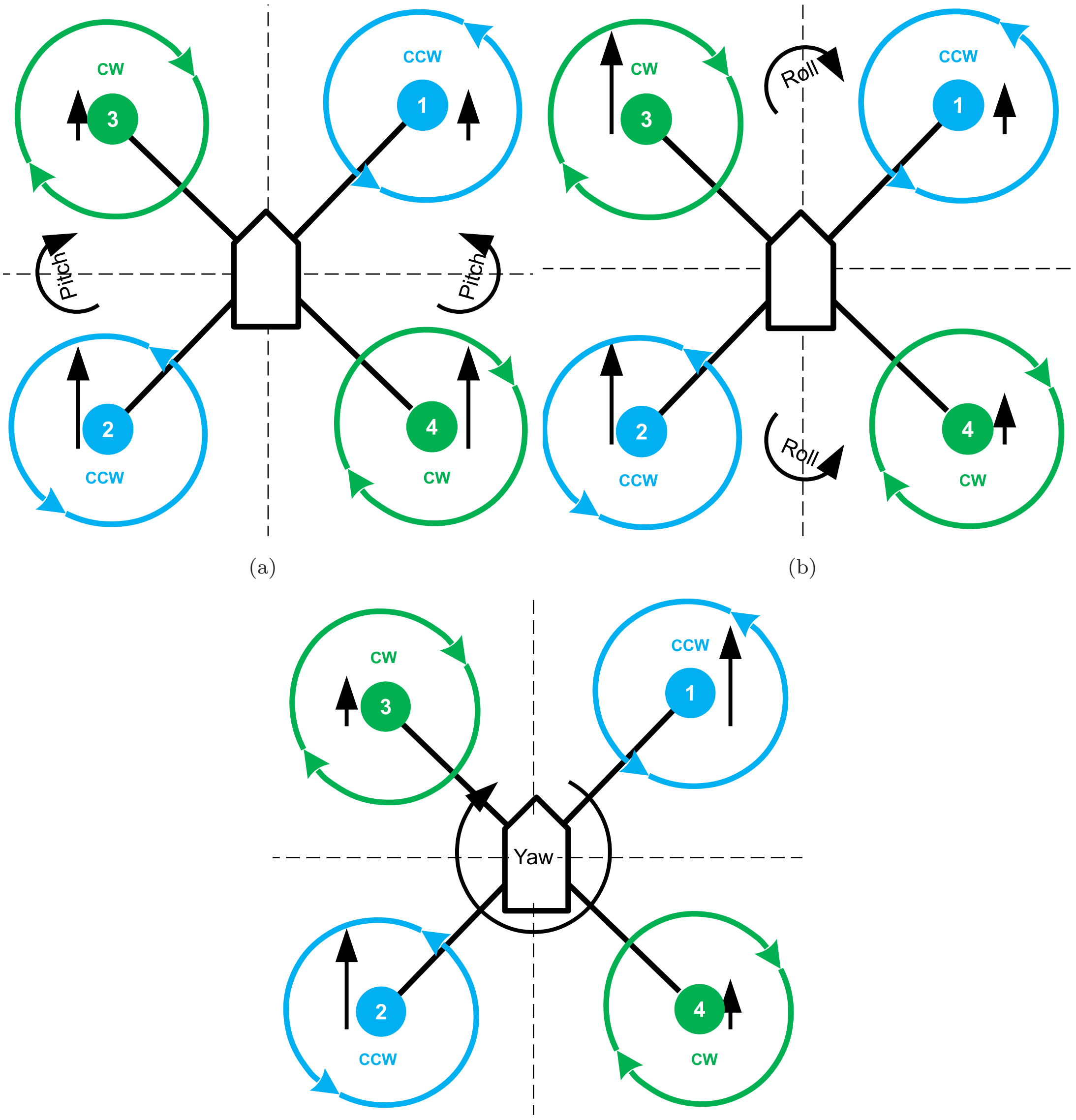

(a) (b) (c)

Figure 2.4: Quadcopter X configuration's flight principles: (a) pitch control; (b) roll control; (c) yaw control

2.1.2.1 Main Components

All the lift that is required for the platform to fly comes from the static thrust generated by the propellers. Because of that brushless outrunner motors are used in multirotors which are fast, strong and durable. Moreover motor and prop combination directly affects lifting capacity and flight time. We have 2 different motors smaller, 62gr, 850KV (KV= 10^3 rpm per volt) one and a bigger, 72gr, 880KV one ,and two different size propellers 10″x4,5″ and 12″x4,5″. Smaller size

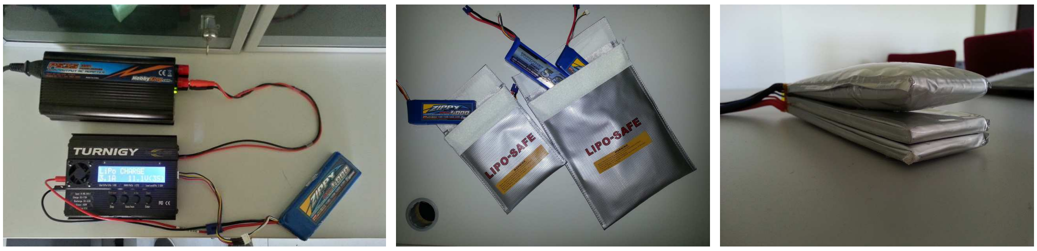

(a) (b) (c)

Figure 2.5: (a)up-left: DC power supply, down-left: LiPo balance charger, down-right: LiPo battery; (b)fireproof charging and storage bags with LiPo batteries; (c) one cell damaged LiPo battery

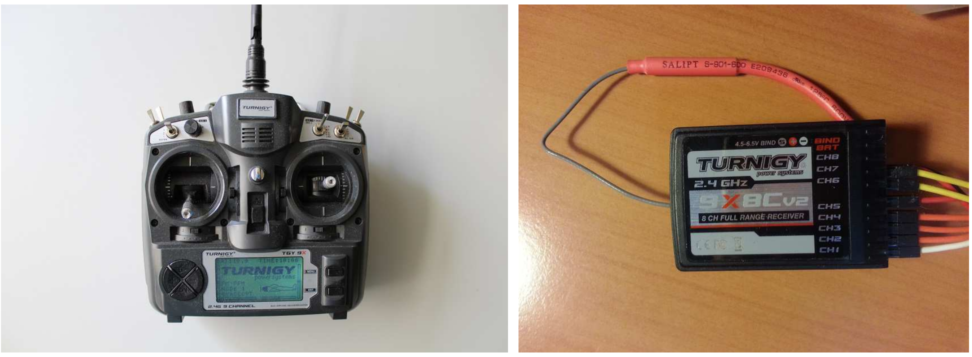
850KV motor with 10″x4,5″ prop is the most efficient setup with 970gr maximum thrust while bigger 880KV motor with 12″x4,5″ prop gives 1380gr maximum thrust for each rotor.

To drive brushless motors, ESCs are used. ESCs are simply high speed step motor drivers that take PWM signal as input. PWM signals generated by the control card are used to determine the speed of the motors.

Selection of battery is one of the key important aspects of a multirotor setup. Lithium polymer batteries (Figure 2.5) are commonly used as they give the best energy density (power/weight ratio) among all known batteries but LiFePo4[22] batteries are also used because of fast charging capabilities, lower discharge voltage drop and longer life. However LiPo batteries should be used with caution such that: they should never be over charged; over discharged; special chargers (Figure 2.5(a)) should be used to balance each cell properly; should be stored carefully preferably in a fireproof LiPo bags(Figure 2.5(b)) and damaged or puffed batteries (Figure 2.5(c)) must never be used, in case of misuse there is a high risk of explosion and fire damage. We have chosen to use 3 cell-series LiPo batteries as they provide around 12 volts as other components like FitPC, downlink system and Kinect also requires 12V supply voltage. Specifications of the batteries that we used can be seen in Table 2.1. We mostly use Zippy 4000 mAh as it has maximum capacity/weight ratio, and Turnigy 5000 mAh as having the biggest capacity.

Table 2.1: LiPo specifications

Brand	Capacity	weight	Discharge Rate
Zippy	3000 mAh	236 gr	20C
Zippy	4000 mAh	300 gr	20C
Turnigy	1000 mAh	90 gr	20C
Turnigy	3300 mAh	295 gr	30C
Turnigy	4000 mAh	350 gr	30C
Turnigy	5000 mAh	410 gr	25C
Raiden	2000 mAh	200 gr	25C

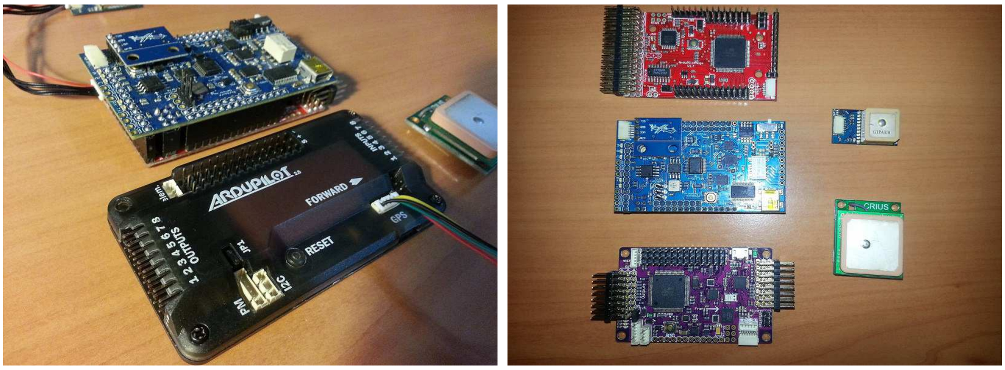


(a) (b)

Figure 2.6: (a)Turnigy 9X Transmitter; (b)Turnigy 9X8C-V2 8-channel receiver

To manually control the quadcopter and as a safety precaution while in autonomous mode, an RC transmitter and receiver with at least 6 channels are required. In addition, as we explain later the on board flight control software we use requires RC transmitter to be used in arming/disarming of the motors. We acquired a Turnigy 9X Transmitter[21] with Turnigy RF9X-V2 module and LCD backlight kit as transmitter (Figure 2.6(a)) ,and bind it with Turnigy 9X8C-V2 8-channel receiver(Figure 2.6(b))

The control card is the brain of the quadcopter. The data obtained from inertial measurement unit (IMU) are processed at the control card to achieve the stability. For the control card, we acquired Arduino-compatible [23] Ardu Pilot Mega with Atmega 1280 processor (APM V1) and Ardu pilot Mega with Atmega 2560 processor (APM V2.5, Figure 2.7). Both cards have a 16MHz processor however APM V1 has 128KB of flash program memory while APM V2.5 has 256KB of flash program memory. The main difference is APM V1 does not have any sensors



(a) (b)

Figure 2.7: (a) Up: APM V1 with IMU oilpan, down: APM V2.5 with case; (b) Up: APM V1, middle: IMU oilpan, down: APM V2.5

and requires an IMU oilpan whereas APM V2.5 has all the sensors embedded. Also the sensors differ from card to card but they still measure the same data with different resolution and accuracy. Available sensors are: 3-axis gyro, accelerometer and magnetometer, along with a high-performance barometer, GPS also comes with the cards but it is off-board.

After uploading the firmware which will be mentioned, later and assembling the components quadcopter is ready to fly via remote control.

2.1.2.2 Secondary Components

In this section components that are optional will be mentioned. Depending on the mission these components might be necessary.

Although the APM boards has an barometer and GPS to measure altitude, an ultrasonic range finder (i.e. sonar, Figure 2.8(a)) is useful especially when hovering below 10m. The sonar sensor generates and transmits a high frequency sound wave and measures the duration of the reflected signal to return. Using the elapsed time and speed of sound in air, the distance is calculated. For low altitude hovering with APM V1 the sonar is necessary. However the barometric sensor in APM V2.5 has a higher resolution and accuracy that the use of sonar is not needed for altitude reading.

(a) (b)

Figure 2.8: (a) Maxbotix LV-MaxSonar-EZ0 Sonar; (b) 3DR radio kit 433MHz

APM boards can be connected to a PC via an ordinary USB cable. On the other hand, if a ground station wanted to be used, APM board and the computer should communicate wirelessly. For this purpose a telemetry kit like Xbee or 3DR radio kit can be used. We used 433MHz 3DR radio kit (Figure 2.8(b)) with baudrate 57600.

As mentioned before LiPo batteries must be used with caution. Discharging the battery below 3V per cell might damage the batteries and a high voltage drop will happen below this level. Motors on the quadcopter will shut down due to the sudden change in the supply voltage, which may result in a crash and serious damage to the quadcopter if it was flying in high altitude. ESCs have a programmable soft cutoff voltage that protects the batteries and reduces the crash damage. Even a safe landing of the quadcopter might be possible as the throttle is reduced softly. Also unbalanced cell voltages might cause a failure of the operation. Because of that each cell's voltage should be monitored and balanced rather than total voltage. Due to these reasons we use an external battery monitor to measure each cell's voltage and sounds an audible warning and flashes a bright LED when it is less than the programmed low voltage value.

2.1.2.3 Componenets for Autonomous Landing

A lightweight computer (FitPC2 Figure 2.9 [24]) with Intel Atom Z530 working at 1.GHz and 1GB of memory is installed on the quadcopter for onboard image processing, data set gathering etc. 4 USB ports allows us to connect APM, Kinect

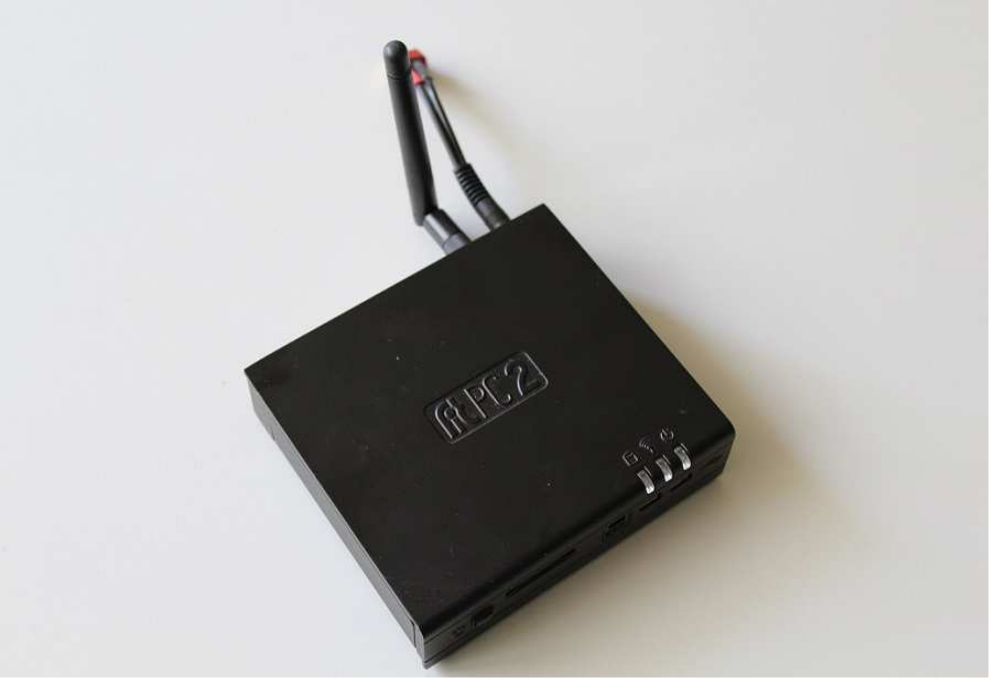


Figure 2.9: FitPC2

and any other required USB devices, also the integrated wireless adapter allows us to connect it remotely. The computer has a built in voltage regulator and can be powered by the 3 cell series Li-Po that powers the quadcopter. However we found out that the computation capabilities of the processor may not be enough for using Kinect in the highest frame rates.

Kinect[25] is the one the key components in our system since our methods rely on depth information. Recent introduction of RGB-D sensors (Red-Green-Blue-Depth) allowed affordable and easy computation of depth maps. Microsoft Kinect is a pioneer of such sensors which was initially marketed as a peripheral input device for computer games. It integrates an infrared (IR) projector, a RGB camera, a monochrome IR camera, a tilt motor and a microphone array. The device can be used to obtain 640x480 pixel depth map and RGB video stream at a rate of 30fps.

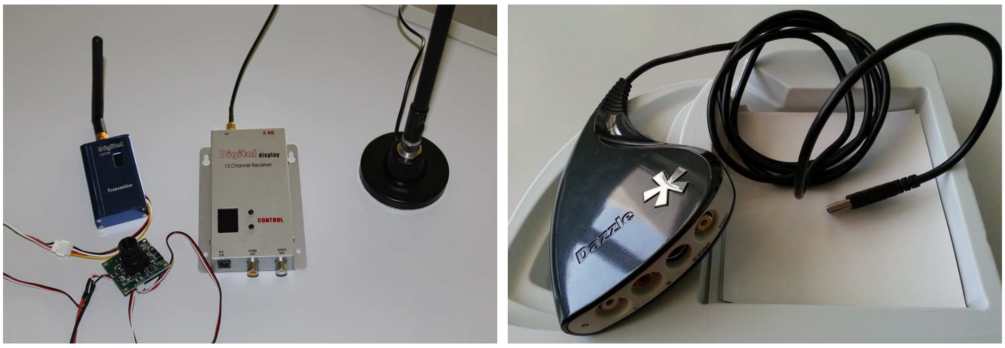
Kinect uses an IR laser projector to cast a structured light pattern to the scene. Simultaneously, an image of the scene is acquired by a monochrome CMOS camera. The disparities between the expected and the observed patterns are used to estimate a depth value for each pixel. Kinect works quite well for indoor environments. However, the depth reading is not reliable for regions that are far more than 5 meters; at the boundaries of the objects because of the shadowing; reflective or IR absorbing surfaces; and at the places that are illuminated directly by sunlight which causes IR interference. Accuracy of the sensor under different conditions were studied in [26–28].

Kinect requires 12V supply voltage and very intolerant to voltage changes. Because of that a voltage regulator placed between quadcopter's LiPo battery and Kinect ensuring stable 12V supply voltage. For Chapter 3 a forward facing placement is adequate while for Chapter 4 placement is perpendicular to the ground.

2.1.2.4 Useful Payload

In an aircraft a useful payload is any extra equipment carried to the something specific to the mission. This can be a camera for inspection mapping mission whereas a fire extinguisher might be payload for fire intervention missions.

Other than the cameras used for capturing high resolution videos or image processing purposes, we wanted to get a live stream video feedback from the quad to see the point of view. In order to do that a 1/3″ CCD camera with a downlink system (Figure 2.10(a)) is installed on the platform. We have tried two different analog video transmitters, a 800mW, 900MHz one and a 1000mW, 2.4GHz one. Each one working with 12V supply voltage making it compatible with our main battery. In open field 2.4GHz transmitter gives better stream, while in closed environments especially filled with Wi-Fi access points both systems give noisy streams. In order to view live stream and be able to record the video a video capture card is used at the ground station (Dazzle, Figure 2.10(b)). First thought was to record the streamed video on the ground but with respect to the findings streamed video was used just for viewing and inspection purposes.



(a) (b)

Figure 2.10: (a) 1000mW 2.4GHz video transmitter, 12 channel receiver, 1/3″ CCD camera; (b) Dazzle video capture card

Another problem about installing a camera to quadcopter platform is that, pitch and roll angles of the quad change in order to change the global position. A fixed camera installment gives too much movement distortion due to continuous changing pitch and roll angles. To obtain a more stable video, a two axis camera gimbal is installed (Figure 2.11). Gimbal's tilt and roll positions are changed with respect to the pitch and roll angle measurements obtained from IMU, making the inverse movements of the quadcopter platform so that camera position stays still with respect to the ground.

2.1.3 Overall System

Complete layout of the system can be seen in Figure 2.12 and our ready to fly quadcopters can be seen in Figure 2.13. Payloads might be added or removed with respect to the mission. As long as lift capacity of the quadcopter is not exceeded, another battery can be added for longer flight times. As the quadcopter is a flying platform the weight of the complete system is important, a heavy configuration will result bad flying times, even not able to fly if the lifting capacity is exceeded. Also the placement of the components is important; the center of the gravity of the system should be at the center and below the rotor height for stability. Our systems overall weight can be seen from Table 2.2

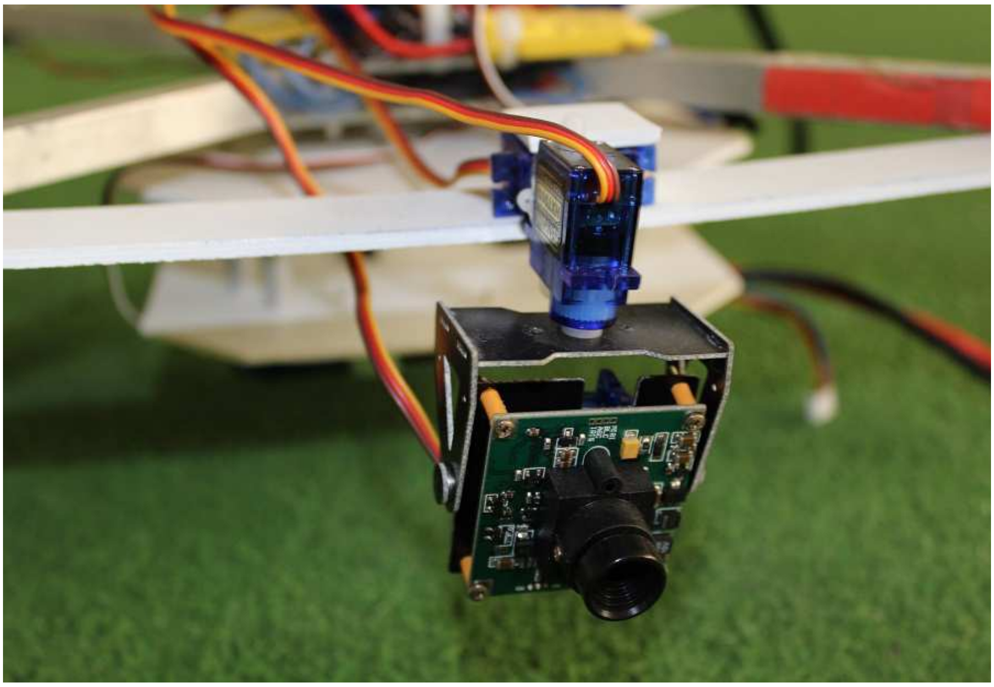


Figure 2.11: Gimbal system and 1/3″ CCD camera

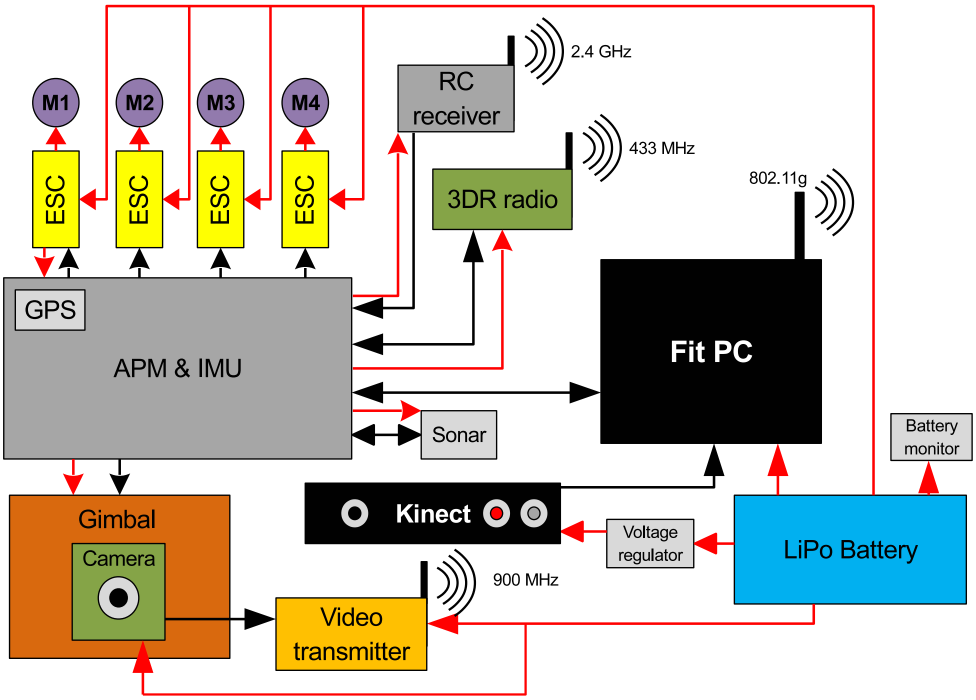
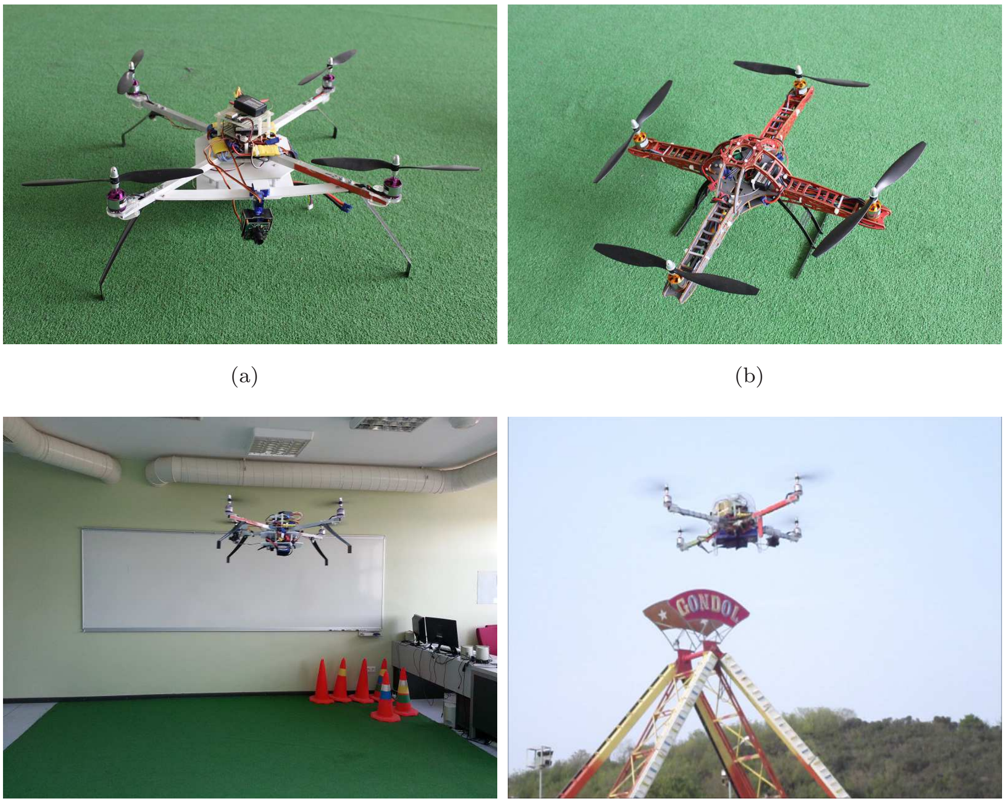


Figure 2.12: Complete system layout



(a) (b) (c) (d)

Figure 2.13: Our quadcopters:(a) Jdrones frame; (b) HobbyKing frame; (c) quad flying indoors; (d) quad flying outdoors at Işık Spring Fest

Table 2.2: Complete system weight

Component	Weight (gr)	Quantity	Total Weight (gr)
Main frame	490	1	490
Props	10	4	40
Motors	62	4	248
ESC	20	4	80
Receiver	24	1	24
APM	55	1	55
Battery	415	1	415
Sonar	10	1	10
Battery monitor	22	1	22
Telemetry	14	1	14
Camera	20	1	20
Gimbal	30	1	30
FitPC	412	1	412
Downlink transmitter	90	1	90
Kinect	450	1	450
Overall System			2400

2.2 Software System

There are two parts of the software that used in quadcopter setup which are the onboard software (i.e. firmware) and the ground control station software. For communication between the APM and PC MAVLink protocol is used.

2.2.1 Onboard Software

Arducopter, an open source project, offers almost a complete UAV solution for different multirotor frames and helicopters[16]. Arducopter originally was an extension to the Ardupilot project started in May 2010. It become a main project in December 2010 and first beta codes released in May 2011. Since then we are testing and using the Arducopter firmware. Firmware has many flight modes including stable, acrobatic, altitude hold, return to home, loiter and follow me modes. On the other hand the latest Arducopter code is not compatible with APM V1 as the flash memory of the board is not sufficient. Latest code for APM V1 is Arducopter 2.3. By removing some of the codes that are unnecessary for our purpose we were able to upload Arducopter 2.6 which has the newer serial communication protocol which will be mentioned in Section 2.2.3. Arducopter 2.9.1 is the latest firmware and can be uploaded to APM V2.5. The firmware continuously reads IMU and make corrections to be able to fly continuously. Attitude control is achieved by a nested PI ->PID loop. In the control mechanism, desired rate of angular rotation is compared that to the raw gyro output in the inner PID loop. The difference is fed back into PID controller and motors speed change to correct the rotation. Desired rate of angular rotation is generated by outer PI loop. The input for this loop is either user input or autopilot input. As an example to the control mechanism, flow chart of roll control is shown in Figure 2.14. Tuning the inner PID loop directly affects the stability whereas the outer PI loop affects mostly the style of flying (fast or slow response).

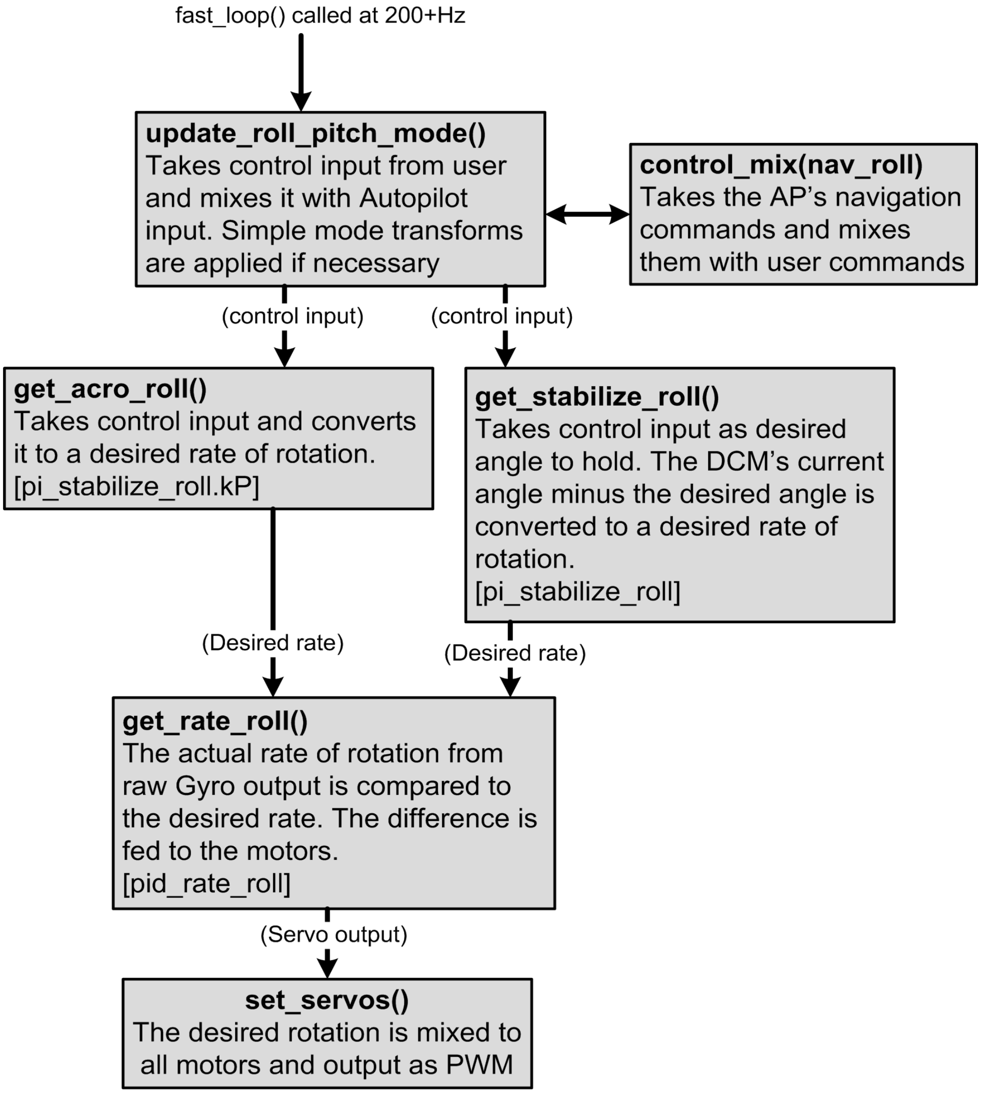

fast_loop() called at 200+Hz
update_roll_pitch_mode()
Takes control input from user and mixes it with Autopilot input. Simple mode transforms are applied if necessary
control_mix(nav_roll)
Takes the AP's navigation commands and mixes them with user commands
(control input)
(control input)
get_acro_roll()
Takes control input and converts it to a desired rate of rotation.
[pi_stabilize_roll.kP]
get_stabilize_roll()
Takes control input as desired angle to hold. The DCM's current angle minus the desired angle is converted to a desired rate of rotation.
[pi_stabilize_roll]
(Desired rate)
(Desired rate)
get_rate_roll()
The actual rate of rotation from raw Gyro output is compared to the desired rate. The difference is fed to the motors.
[pid_rate_roll]
(Servo output)
set_servos()
The desired rotation is mixed to all motors and output as PWM

Figure 2.14: Flow chart of roll control [16]

2.2.2 Ground Control Station

UAVs require ground control stations (GCSs) for mission planning, manual controlling or for observation purposes. For MAVs there are many open source GCS software and some of the compatible ones with APM board and Arducopter software are APM planner, HK GCS, QGroundControl. We are using APM planner which is created by Michael Oborne[16] and does a lot more than a GCS (Figure 2.15). First of all, the firmware of APM can be updated using the graphical user interface (GUI), compiled firmware automatically downloads from the server and uploads. The on board software configuration can be done in the most comprehensive way with a user friendly GUI. Raw sensor data can be obtained which can be used to detect any vibration or sensor measurement error. Missions can be created on Google Maps and can be uploaded to the APM board. By using a telemetry kit, real time data coming from the quad can be seen; instead of offline mission planning commands can be given on the fly; and last but not least instead of a RC transmitter a USB joystick can be used as a controller via Mission planner.

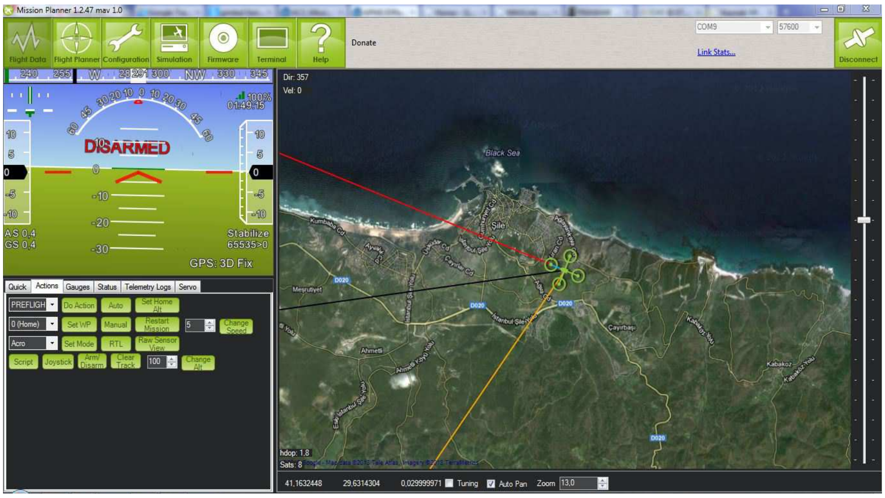
2.2.3 MAVLink

MAVLink is an open source communication protocol with lightweight serialization functions for microcontrollers [30]. It is a header only message marshaling library. It allowed different GCS software to be reusable for different platforms where it is integrated. Some of the autopilots that use MAVLink are: ArduPilotMega[31], pxFMU Autopilot[32] and SLUGS Autopilot[33]; some of the GCS software that use MAVLink are: QGroundControl[34], HK GCS[35], APM Planner[29]. Also the open source Robot Operating System(ROS, [36]) developed by Stanford Artificial Intelligence Laboratory, is compatible with MAVLink protocol[37].

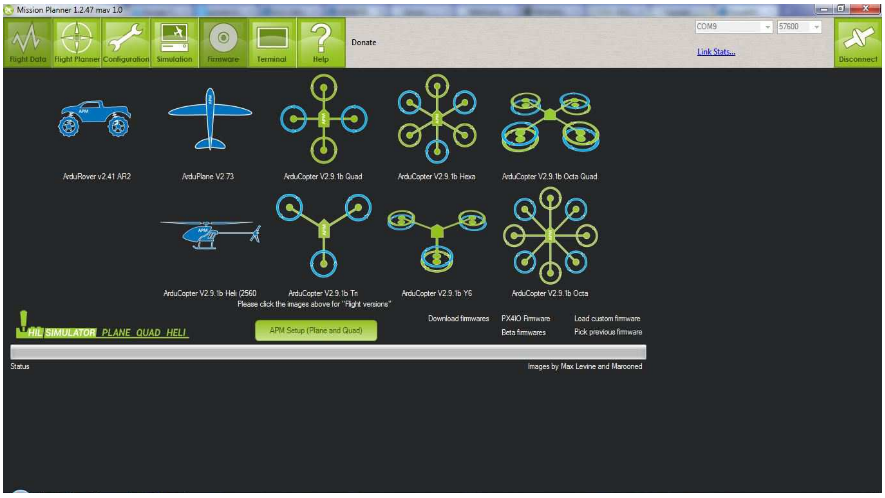
2.2.4 RAVLAB Control Station

It is possible to write a program that uses MAVLink protocol to give commands to quadcopter. It is important to understand the structure of MAVLink commands and firmware capabilities. To obtain the command of the quadcopter some steps are necessary, which can be simplified as:

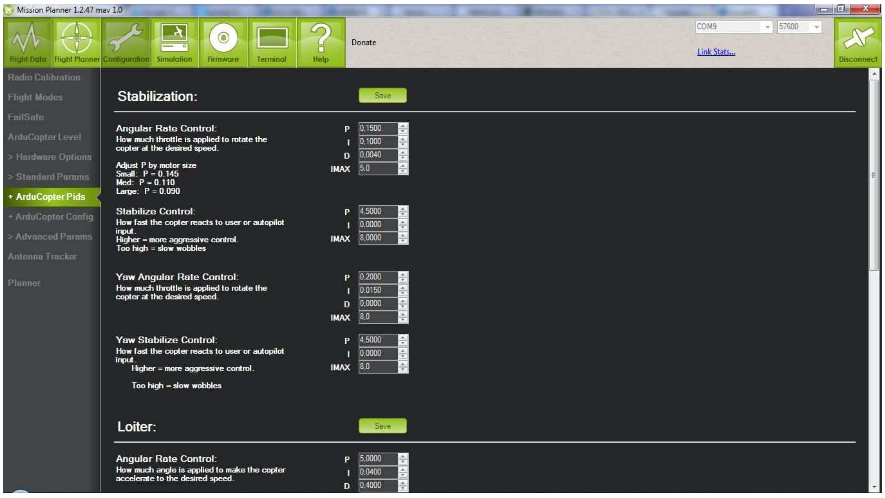
1. Find the system state: MAVLink protocol uses a periodic signal called as “heartbeat” to show that the system is alive and working. First to understand the condition of the system periodic heartbeat signal should be



(a)



(b)



Stabilization:

Angular Rate Control:
How much throttle is applied to rotate the copter at the desired speed.

Adjust P by motor size
Small: P = 0.145
Med: P = 0.110
Large: P = 0.090

P 0,1500
I 0,1000
D 0,0040
IMAX 5,0

Stabilize Control:
How fast the copter reacts to user or autopilot input.
Higher = more aggressive control.
Too high = slow wobbles

P 4,5000
I 0,0000
IMAX 8,0000

Yaw Angular Rate Control:
How much throttle is applied to rotate the copter at the desired speed.

P 0,2000
I 0,0150
D 0,0000
IMAX 8,0

Yaw Stabilize Control:
How fast the copter reacts to user or autopilot input.
Higher = more aggressive control.

Too high = slow wobbles

P 4,5000
I 0,0000
IMAX 8,0

Loiter:

Angular Rate Control:
How much angle is applied to make the copter accelerate to the desired speed.

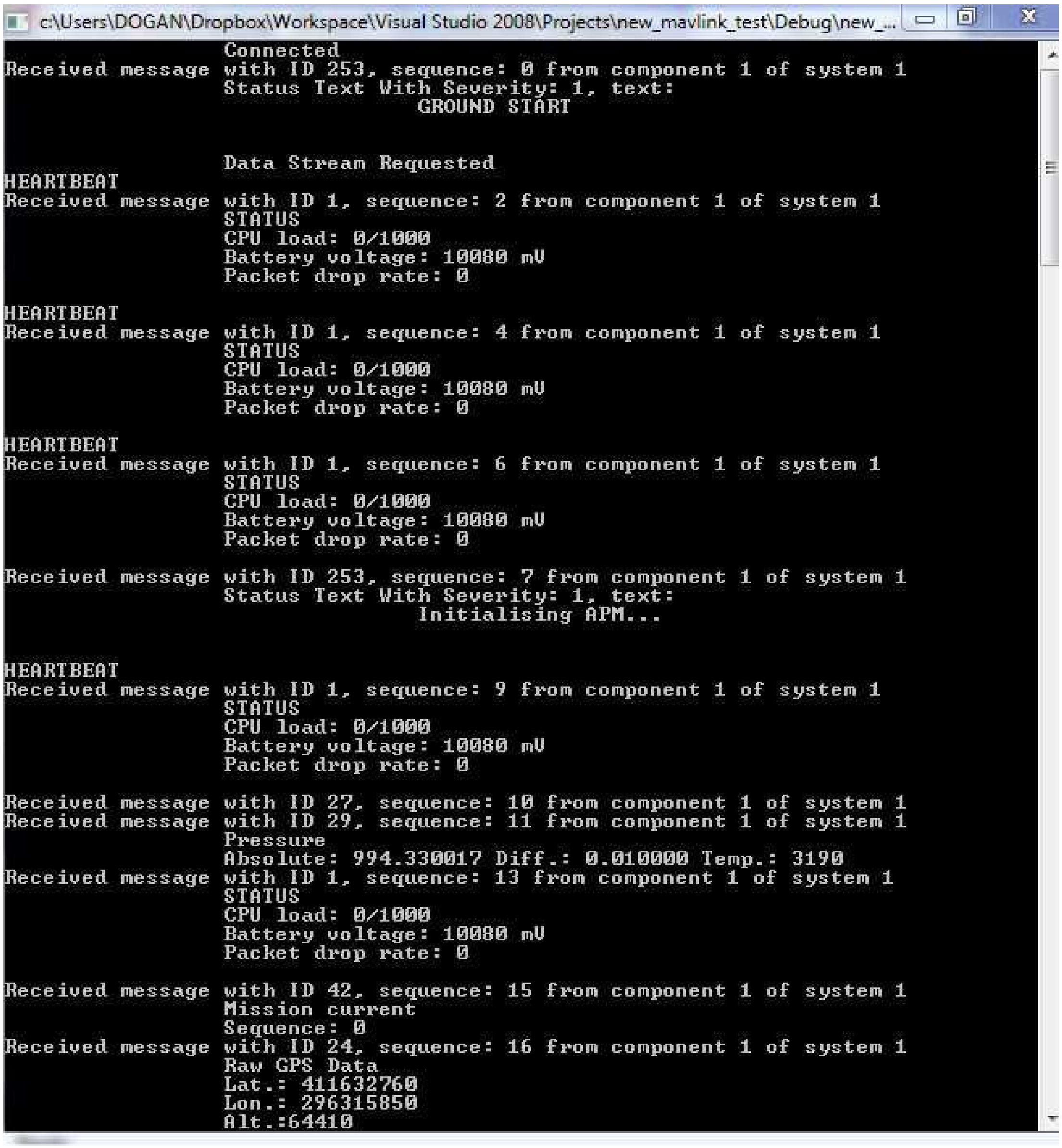
P 5,0000
I 0,0400
D 0,4000

(c)

Figure 2.15: APM planner screens:(a) Flight data screen; (b) firmware selection screen; (c) PID parameters tuning screen[29]

decoded which has the information of if the quadcopter is booting, idle, landing or flying.

2. Request data stream: The heartbeat signal has limited information about the system, for more information a data stream request command should be used.
3. Decode data stream: Returning messages from data stream request command should be decoded.
4. Arm motors: When powered up, to prevent unwanted reaction from the motors due to the manual movements of the platform like relocating for take-off, system stays in disarmed condition. To arm the motors, full right rudder RC input with zero throttles should be given for 2 seconds or using MAVLink protocol, arm motors command should be given. When the motors are armed APM records to current GPS position as home position, if the GPS information is not available the first position is taken when GPS information is available.
5. Take-off: Take-off command with desired altitude can be given after arming the motors.
6. Change system state: While hovering in the air drifting in a direction is common. To ensure hovering in the same spot system state should be changed to loiter if GPS is available. For indoor environment an optical flow sensor, which our system does not have, is required to obtain same position.
7. Go to GPS position: To move the quadcopter to a new position two different command type can be used. First a GPS position can be given to APM and the firmware will give required commands to go to the position.
8. Move position: Secondly, by using RC channel override command, quadcopter's position can be changed as it is controlling by an RC remote, thus this is the only way if the environment is indoors.
9. Land: The last step is to land, again using MAVLink's land command. APM changes its state to disarm after a successful landing.


c:\Users\DOGAN\Dropbox\Workspace\Visual Studio 2008\Projects\new_mavlink_test\Debug\new_...

```
                 Connected
Received message with ID 253, sequence: 0 from component 1 of system 1
                 Status Text With Severity: 1, text:
                               GROUND START


                 Data Stream Requested
HEARTBEAT
Received message with ID 1, sequence: 2 from component 1 of system 1
                 STATUS
                 CPU load: 0/1000
                 Battery voltage: 10080 mV
                 Packet drop rate: 0

HEARTBEAT
Received message with ID 1, sequence: 4 from component 1 of system 1
                 STATUS
                 CPU load: 0/1000
                 Battery voltage: 10080 mV
                 Packet drop rate: 0

HEARTBEAT
Received message with ID 1, sequence: 6 from component 1 of system 1
                 STATUS
                 CPU load: 0/1000
                 Battery voltage: 10080 mV
                 Packet drop rate: 0

Received message with ID 253, sequence: 7 from component 1 of system 1
                 Status Text With Severity: 1, text:
                               Initialising APM...


HEARTBEAT
Received message with ID 1, sequence: 9 from component 1 of system 1
                 STATUS
                 CPU load: 0/1000
                 Battery voltage: 10080 mV
                 Packet drop rate: 0

Received message with ID 27, sequence: 10 from component 1 of system 1
Received message with ID 29, sequence: 11 from component 1 of system 1
                 Pressure
                 Absolute: 994.330017 Diff.: 0.010000 Temp.: 3190
Received message with ID 1, sequence: 13 from component 1 of system 1
                 STATUS
                 CPU load: 0/1000
                 Battery voltage: 10080 mV
                 Packet drop rate: 0

Received message with ID 42, sequence: 15 from component 1 of system 1
                 Mission current
                 Sequence: 0
Received message with ID 24, sequence: 16 from component 1 of system 1
                 Raw GPS Data
                 Lat.: 411632760
                 Lon.: 296315850
                 Alt.:64410
```

Figure 2.16: RAVLAB Control Station screenshot

At the beginning of this thesis we wanted to get the full control of the quad but we decided to follow an alternative approach afterwards. But we have written a little program that connects to the quadcopter, understand the heartbeat message, requests data stream from the APM, and decodes the received messages which can be GPS data or IMU readings etc. (Figure 2.16). Possible future work is discussed in Conclusion.

Chapter 3

Ground Plane Detection Using an RGB-D Sensor

Regardless of the method or the device that is used to obtain depth information there are several works which approach to the ground plane detection problem based on the relationship between a pixel's position and it is disparity [17, 38–42]

Li *et al.* show that the vertical position (y) of a pixel of the ground plane is linearly related to its disparity $D(y)$ such that one can seek a linear equation $D(y) = K1 + K2 * y$, where $K1$ and $K2$ are constants which are determined by the sensor's intrinsic parameters, height, and tilt angle. However, ground plane can be directly estimated on the image coordinates using the plane equation based on disparity $D(x, y) = ax + by + c$ without determining mentioned parameters. A least squares estimation of the ground plane can be performed offline (i.e. by pre-calibration) if a ground plane only depth image of the scene is available [39]. Another common approach is to use RANSAC algorithm which allows fitting of the ground plane even the image includes other planes and surfaces [38, 43, 44]. Since RANSAC is used to estimate linear planes, the ground plane is assumed to be the dominant linear plane in the image.

There are some other works of segmentation of the scene into relevant planes [44, 45]. The work of Holz *et al.* clusters surface normals to segment planes and reported to be accurate in close ranges [44].

In [17] row histograms of the disparity image are used to model the ground plane. In the image formed of the row histograms (named as *V-disparity*), the ground plane appears as a diagonal line. This line, which is detected by Hough Transform, was used as the ground plane model.

In this chapter, a novel and simple algorithm is presented to detect the ground plane without the assumption of that it is the largest region. The method is based on the fact that if a pixel is from the ground plane, its depth value must be on a rationally increasing curve placed on its vertical position. However, the degree of this rational function is not fixed due to reasons which will be explained later. Nevertheless, it can be easily estimated by an exponential curve fit which can be used as a ground plane model. Later, the pixels which are consistent with the model are detected as ground plane whereas the others are marked as obstacles. While this is the base model which can be used for a fixed viewing angle scenario, an extension of it, is provided for dynamic environments where sensor viewing angle changes from frame to frame. Moreover, the relation of our approach to the V-disparity approach [17] is noted, which rely on the linear increase of disparity and fitting of a linear line to model the ground plane. Thus, experiments are provided which test and compare both approaches on the same data.

This chapter is organized as follows: In Section 3.1 the proposed method is presented. Section 3.2 presents the results of the experiments. Conclusion and future work are presented in Section 3.3.

3.1 Ground Plane Detection

Presented method consists of two sub-approaches: First, simple base approach which works quite well for a fixed sensor view is explained. However, it requires an image of the ground plane (without any obstacles) to perform a pre-calibration. The second approach is developed for non-fixed sensor viewing angle. It suits better to dynamic environments.

3.1.1 Detection for fixed pitch

In a common scenario, the sensor views the ground plane with an angle (i.e. pitch angle). The sensor's pitch angle (Figure 3.1(a)) causes allocation of more pixels for the closer locations of the scene than the farther parts. So that linear distance from the sensor is projected on the depth map as a rational function. This is demonstrated by an example of the intensity coded depth map image obtained from Kinect (Figure 3.2(a)). Any column of the depth image will show that the depth value increases not linearly but exponentially from bottom to top (i.e. right to left in Figure 3.2(b)).

In this section it is assumed that the sensor's viewing angle is fixed and its roll angle is zero (Figure 3.1(a)). Furthermore, a "ground plane only" depth image will have all columns equal to each other. These columns are estimable by an exponential function.

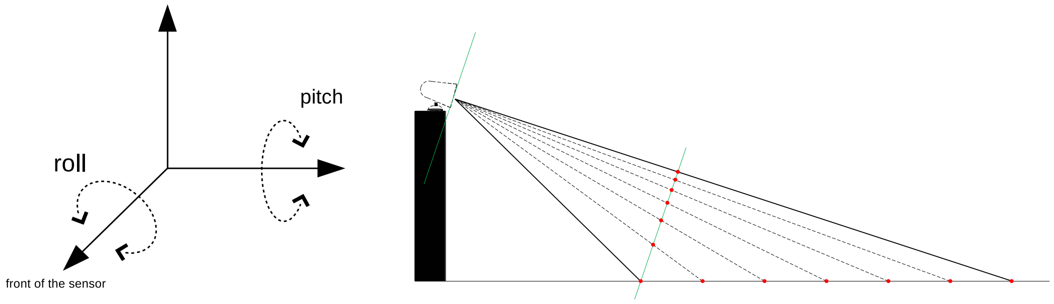
Thus, a curve can be fitted to any (or average) vertical line of the depth map. By trial and error it is found that a good fit is possible with sum of two exponential functions in the following form:

$$f(y) = ae^{by} + ce^{dy} \tag{3.1}$$

, where $f(y)$ is the pixel's depth value and y is the its vertical location (i.e. row index) in the image. The coefficients (a, b, c, d) depend on the intrinsic parameters, pitch angle, and the height of the sensor.

These coefficients are estimated by a least squares fitting method. Then it is possible to reconstruct a curve, which is named here as the *reference ground plane curve* (C_R).

In order to detect ground plane pixels in a new depth map, the columns of the new depth map (C_U) are compared to C_R. Any value that is under C_R represents an object (or any protrusion), whereas values above the reference curve represent



(a) (b)

Figure 3.1: (a) Roll & pitch axis, (b) sensor view pitch causes linearly spaced points to mapped as a exponential increasing function.

drop-offs, holes (e.g. intrusions, downstairs, edge of a table) in the scene. Hence the absolute difference against a pre-defined threshold value T is compared; mark the pixels as ground plane if difference is less than T.

For the comparison depth values which are equal to zero are ignored as they indicate sensor reading errors. The experiments concerning this part are presented in Section 3.2.

3.1.2 Detection for changing pitch and roll

The fixed pitch angle scheme explained above is quite robust. However, it is not suitable for the scenarios where the pitch and roll angles of the sensor changes. Generally the mobile robots exhibit movements on the sensors' platform. Pitch and roll movements can be compensated by using an additional gyroscopic stabilization [46]. However, here a computational solution is proposed. In this approach a reference ground curve from a reference pre-calibration image is not calculated but it is estimated each time from the particular input frame.

A higher pitch angle (sensor almost parallel to the ground) will increase the slope of the ground plane curve. Whereas a non-zero roll angle (horizontal angular change) of the sensor forms different ground plane curves along columns of the depth map (Figure 3.2(c)). Such that at one end the depth map exhibits curves of higher pitch angles while towards the other end we observe curves of lower pitch angles. These variations complicate the use of a single reference curve for that particular frame.

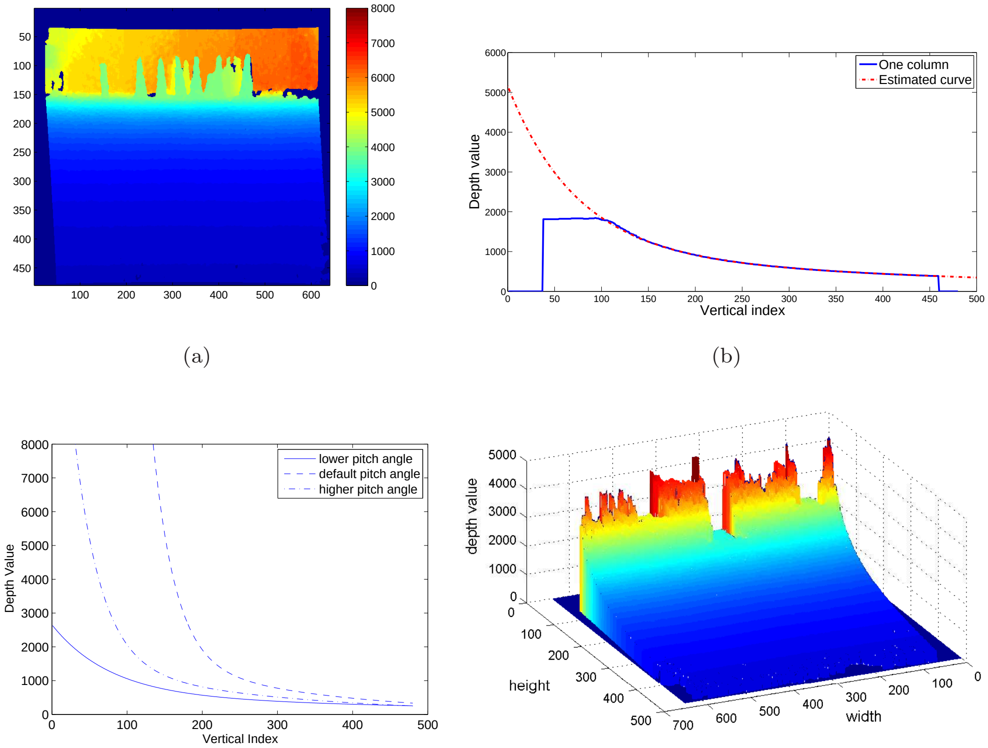
To overcome roll angle affects presented approach aims to rotate the depth map to make it orthogonal to the ground plane. If the sensor is orthogonal to the ground plane it is expected to produce equal or very similar depth values along every horizontal line (i.e. rows). And this similarity can be simply captured by calculating a histogram of the row values such that a higher histogram peak value indicates more similar values along a row. Let h_r shows the histogram of the rth row of a depth image (D) of R rows, and let us denote the rotation of depth image with D_θ.

$$argmax_\theta(\sum_{r=1}^{R} argmax_i(h_r(i, D_\theta))) \tag{3.2}$$

Thus for each angle value θ in a predefined set, the depth map is rotated with an angle θ and the histogram h_r is computed for every row r. Then, the angle θ that gives the total maximum peak histogram value (summed over rows) is estimated as the best rotation angle. This angle is used to rotate the depth map prior to the ground plane curve estimation. After the roll affect is removed the pitch compensation curve estimation scheme can start.

As explained, changes of pitch angle create different projection and different curves (Figure 3.2(c)). Moreover, since the scene may contain obstacles we must define a new approach for ground plane curve estimation.

In a scene that consists of both the ground plane and objects, as in Figure 3.2(d), maximum value along a particular row of the depth map must be due the ground plane, unless an object is covering the whole row. This is because the objects that are closer to the sensor than the ground plane surface that they occlude. Therefore, if the maximum value across each row (r) of the depth map (D) is taken, which we name as the *depth envelope* (E), it can be used to estimate the



(a)

(b)

(c)

(d)

Figure 3.2: (a) An example depth map image, (b) one column (y=517) of the depth map and its fitted curve representing the ground plane (the values left to the index of the maximum value (vertical index=150) are excluded from fitting, (c) ground plane curves for different pitch angles, (d) depth map in three dimensions showing the drop-offs caused by the objects.

reference ground plane curve (C_R) for this particular depth frame.

$$E(r) = max_i(D(c_i, r)) \tag{3.3}$$

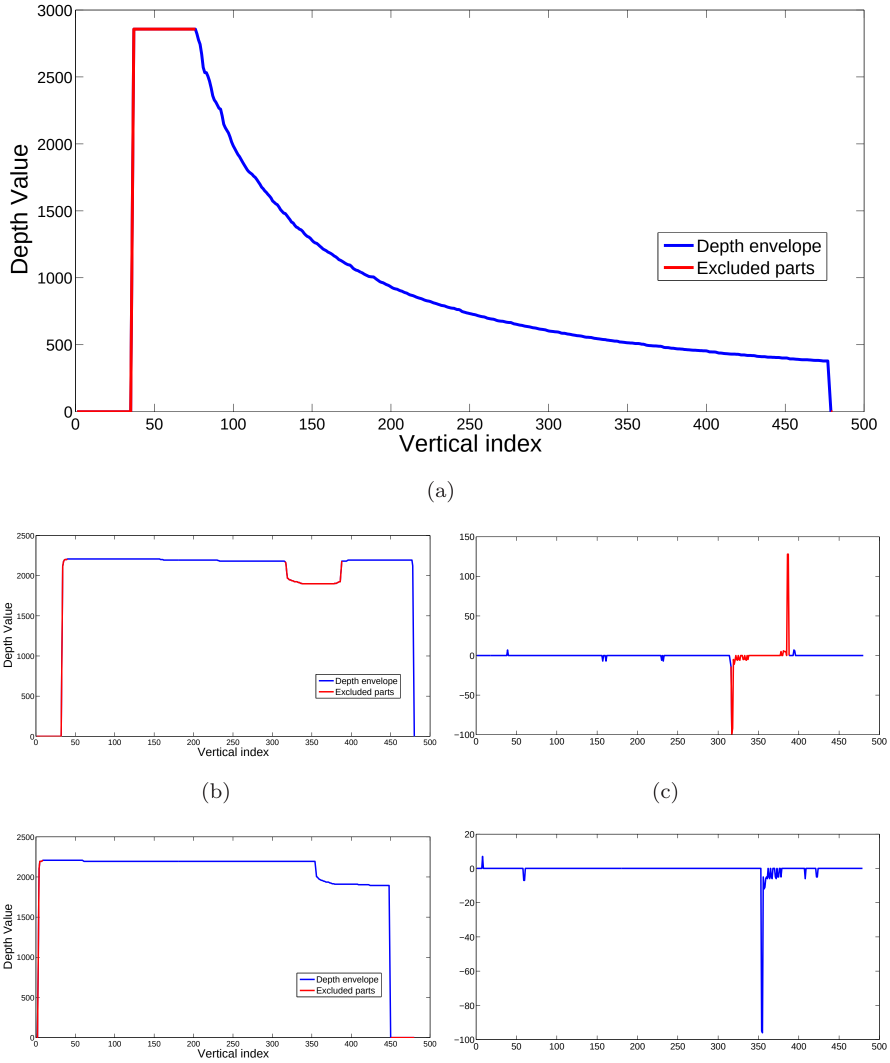
The estimation is again performed by fitting the aforementioned exponential curve (3.1). Prior to the curve fitting median filtering is performed to smooth the depth envelope. Moreover, depth values must increase exponentially from bottom of the scene to the top. However, when the scene ends with a wall or group of obstacles this is reflected as a plateau in the depth envelope. Hence the envelope (E) is scanned from right to left and the values after the highest peak are excluded from fitting as they cannot be a part of the ground plane.

There are two conditions which affect the ground plane curve fit adversely. First, when one or more objects cover an entire row, this will produce a plateau in the profile of the depth map. However, if the rows of the “entire row covering object or group” do not form the highest plateau in the image, ground plane continues afterwards curve continues and the object will not affect the curve estimation.

Second, any drop-offs (e.g. hole, stairs) exhibit higher depth values than the ground plane: drop-offs cause sudden increases (hills) on the depth envelope. If a hill is found on the depth envelope, the estimated curve will be produced by a higher fitting error.

In order to ensure that fitting is not done on objects which can be seen on the depth envelope, an exclusion rule is applied. Depth value increases from bottom of the scene to the top as long as the ground plane continues. Thus, values that comes after the maximum depth value are excluded from left of the depth envelope(Figure 3.3(a)). It is also known that the zero valued pixels are non-reading zones and readings above 5000 are not consistent so these are excluded as well. If a row covering object exists in the scene, it can be seen in the depth envelope as a discontinuity. In order to detect a row covering object first derivative of the depth envelope is taken. Then a negative peak followed by a positive peak is searched which indicates the row covering object (Figure 3.3(c)). Index values between these peaks are excluded (Figure 3.3(b)). However if the row covering object is located at the bottom of the scene (Figure 3.3(d)) it can not be located with this approach (Figure 3.3(e)).

After estimating the ground plane reference curve coefficients for the frame, every column is compared with the reference curve as it was done for the fixed pitch angle algorithm. The pixels are classified as ground plane and non-ground plane by comparing against a threshold T. The value of T was determined by overall accuracy.



(a)

(b)

(c)

(d)

(e)

Figure 3.3: (a) values before the maximum depth value are excluded, (b) row covering object, shown in red, are exculuded, (c) first derivative of (b) is shown, red parts show found row covering object, (d) row covering object at the bottom of the scene, (e) first derivative of (d) is shown, only one peak caused by the row covering object is visible

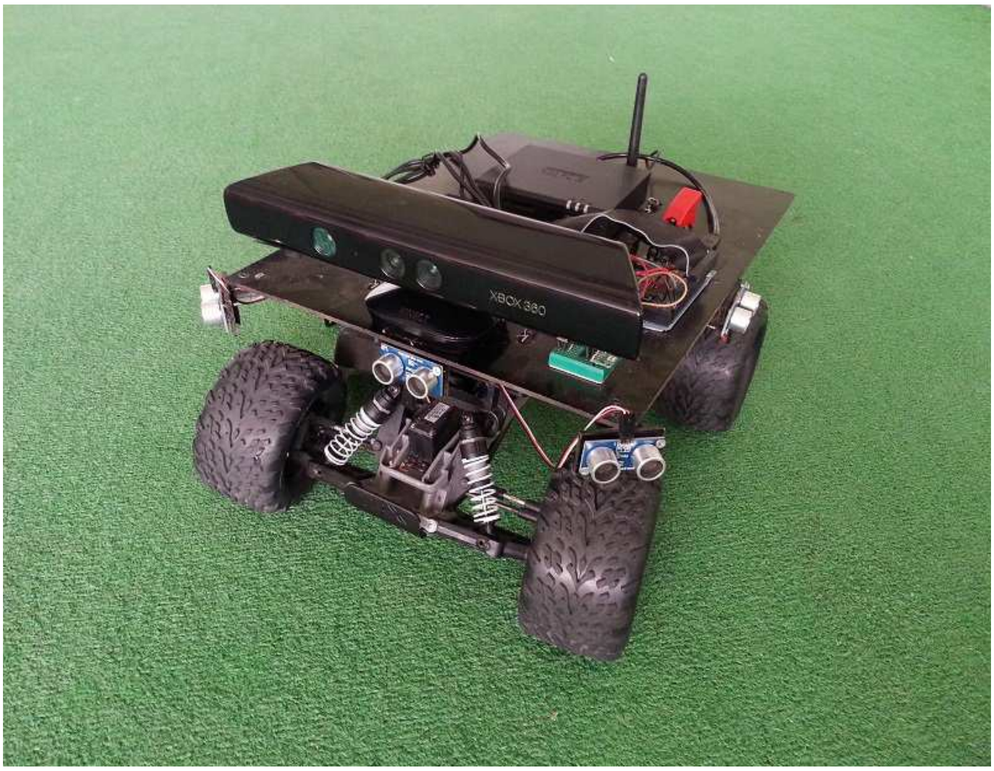


Figure 3.4: Mobile robot platform

3.2 Experiments

The algorithm is tested on four different multi-frame data sets that were not used in the development phase. The dimensions of the depth map and RGB images are 640x480. Two of these datasets (dataset-1 and dataset-2) were manually labeled to provide ground truth and were used in plotting ROC (Receiver Operating Curves), whereas the other two were manually (visually) examined. Dataset-1 and dataset-2 composed of 300 frames captured on a mobile robot platform (Figure 3.4[47]) which moves in the laboratory floor which included many obstacles, different surfaces, and walls. Dataset-3 (300 frames) created with the same platform; however, the pitch and roll angles change excessively. Dataset-4 included 12 individual frames acquired from difficult scenes such as narrow corridors, wall only scenes etc. Three different versions are compared for presented approach: A1-fixed pitch, A2-pitch compensated, A3-pitch and roll compensated. There is only one free parameter for A1 and A2 that is threshold T, which is estimated by

ROC analysis; whereas the 3rd roll compensation algorithm requires pre-defined angle set to search for best rotation angle: $\{-30°,\ -28°,..,+30°\}$. Least squares fit was performed by Matlab curve fitting function with default parameters. However, the depth values which are equal to zero, or above 5000 are excluded due to inaccurate sensor readings. Additionally, as explained previously, for algorithm A2 and A3 the indices positioned to left of the maximum of the column depth value must be excluded from the fits since they do not represent ground plane. Finally, note that A1 requires a onetime pre-calibration and estimation of the coefficients for the reference ground plane curve, whereas A2 and A3 estimate coefficients separately for each new frame.

Moreover, the results are compared with V-disp method [17]. We note that V-disp is originally developed for stereo depth calculation where disparity is available before depth. To implement V-disp method by Kinect depth stream, disparity is calculated from the depth map (i.e. $1/D$), row histograms are calculated to form V-disp image, and then Hough transform is used to estimate ground plane line. We had to put a constraint on the Hough line search in $[-60°, -30°]$ range to have relevant results.

Since A3 and A2 algorithms are same except for the roll compensation, the results of A2 to A1 and V-disp will be examined and compared; however A3 results are compared only against A2 to show the effect of roll compensation scheme.

Figure 3.6(a) and 3.6(b) show ROC curves and overall accuracies plotted for presented fixed and pitch compensated algorithms (A1 and A2) and V-disp method on dataset-2 (dataset-1 is similar). Here, true detection (y-axis) represents the ratio of correctly detected ground plane pixels to all; false detection (x-axis) represents the ratio of ground plane pixels which are misclassified as obstacle pixels to all ground plane pixels. It can be seen that presented pitch compensated algorithm is superior to both V-disp which is better than presented fixed algorithm. However, note that dataset-2 has frames taken in a dynamic environment and the fixed pitch algorithm performs a pre-calibration with one frame and evaluates all frames.

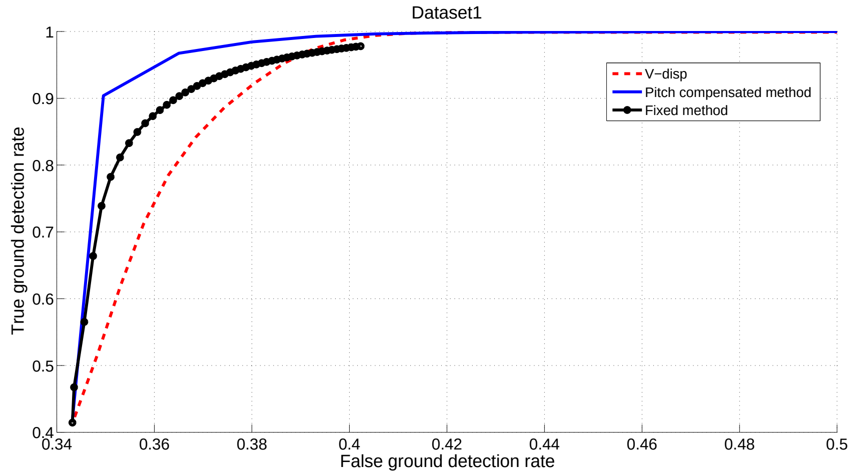
When the best accuracy point thresholds are selected and algorithms are ran on dataset-2, accuracy vs. frames result is obtained(Figure 3.6(c)). In addition curve fitting error is recorded for pitch compensated algorithm (A2). It can be seen that both methods are quite stable with the exception being high curve fitting error frames for A2. It is also easy to spot these frames on live data sequences.

Beside multi-frame datasets, here some example single input-output pairs are included(Figure 3.7 3.8). Here ground plane is marked with black and obstacles were marked with white to ease viewing. In Figure 3.7(a), a cluttered scene is observed. Note that its depth map contained sensor reading errors (zeros) because of the lighting and reflective patches (Figure 3.7(b)). The output of A2 is shown in right column (Figure 3.7(c)). It can be seen that algorithm is quite successful in the regions where there is depth reading. Despite that it is possible to reduce the spurious noisy detections; we show here the raw (not post-processed) outputs.

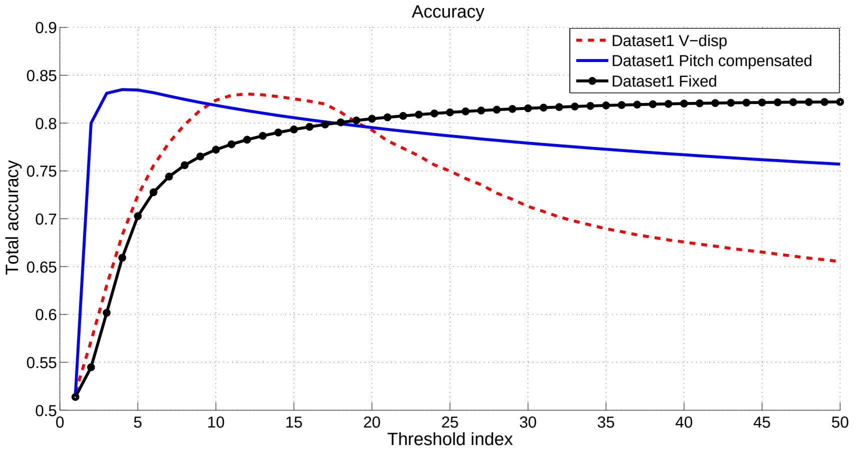
Figure (3.7(d),3.7(e),3.7(f)) show another difficult scene where the robot with sensor is positioned in front of a stairs going down. Due to reflective marble floor the sensor produce many zeros in the close ground plane. In addition, many zeros are observed in distant walls. However, the output is quite successful in the sense that the close plan ground floor and the edge of the stairs is correctly identified.

Despite that dataset-1 and 2 are similar, dataset-3 contains excessive roll changes which were used to test roll compensation (A2 vs. A3). The outputs show that roll compensation is able to detect and correct rotations. Figure 3.7(g)) show one of the frames from dataset-3, where the sensor is rolled almost 20° degrees. Figure 3.7(h),3.7(i) shows the respective outputs of A2 and A3. It can be seen that roll compensation provides a significant advantage if sensor can roll.

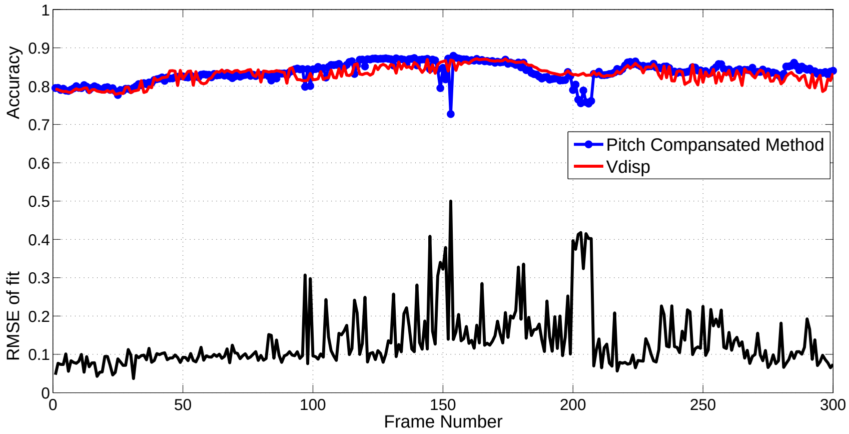
Finally, Figure 3.8 shows two output pairs (overlayed on RGB) for A2 and V-disp. It can be seen that both methods can detect ground planes in scenes where



(a)

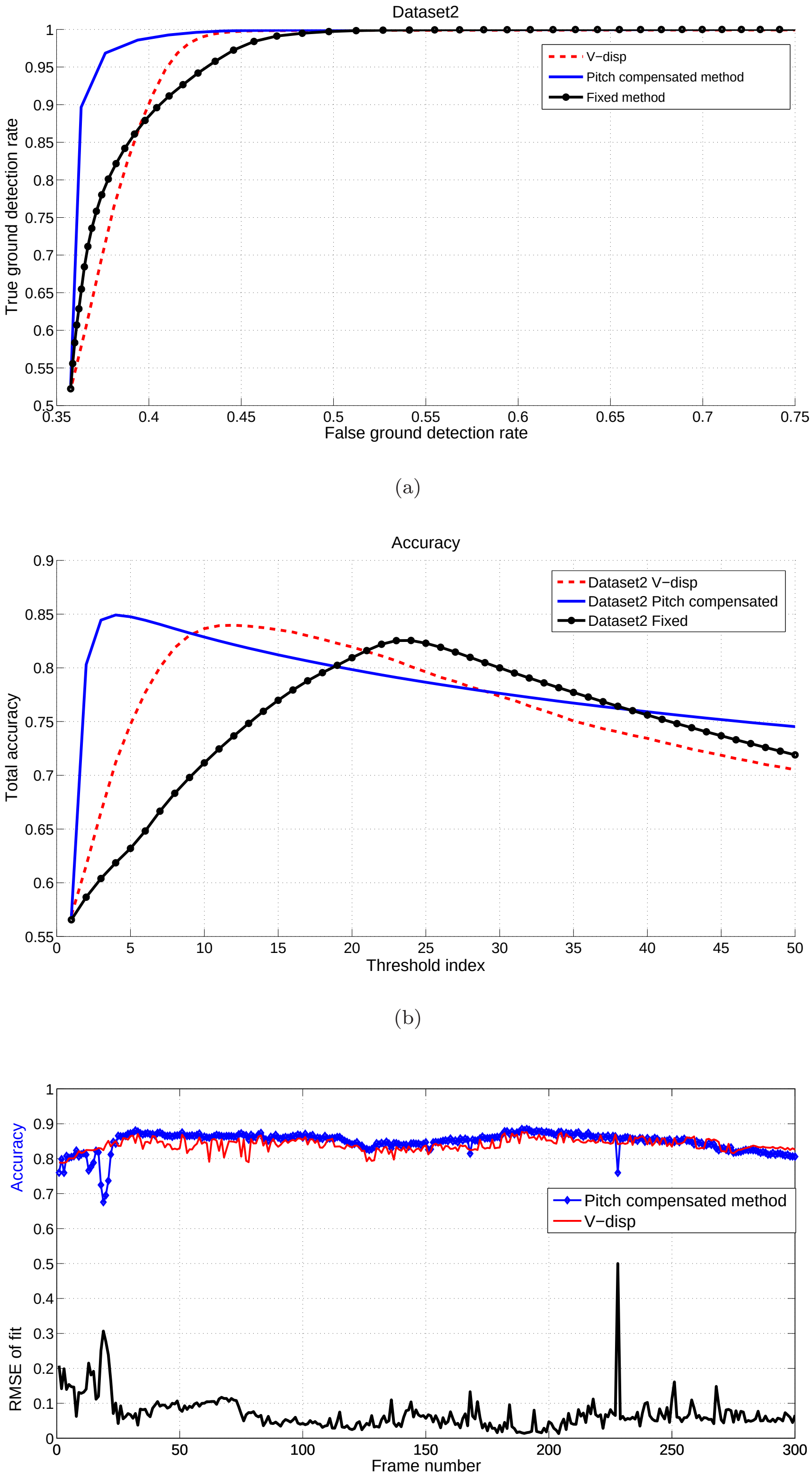


(b)



(c)

Figure 3.5: (a) ROC curves comparing V-disp and our fixed and pitch compensated algorithms (A1-A2) for data set 1, (b) average accuracy over 300 frames vs. thresholds, (c) accuracy and curve fit error of A2 for individual frames.



(a)

(b)

(c)

Figure 3.6: (a) ROC curves comparing V-disp and our fixed and pitch compensated algorithms (A1-A2) for data set 2, (b) average accuracy over 300 frames vs. thresholds, (c) accuracy and curve fit error of A2 for individual frames.

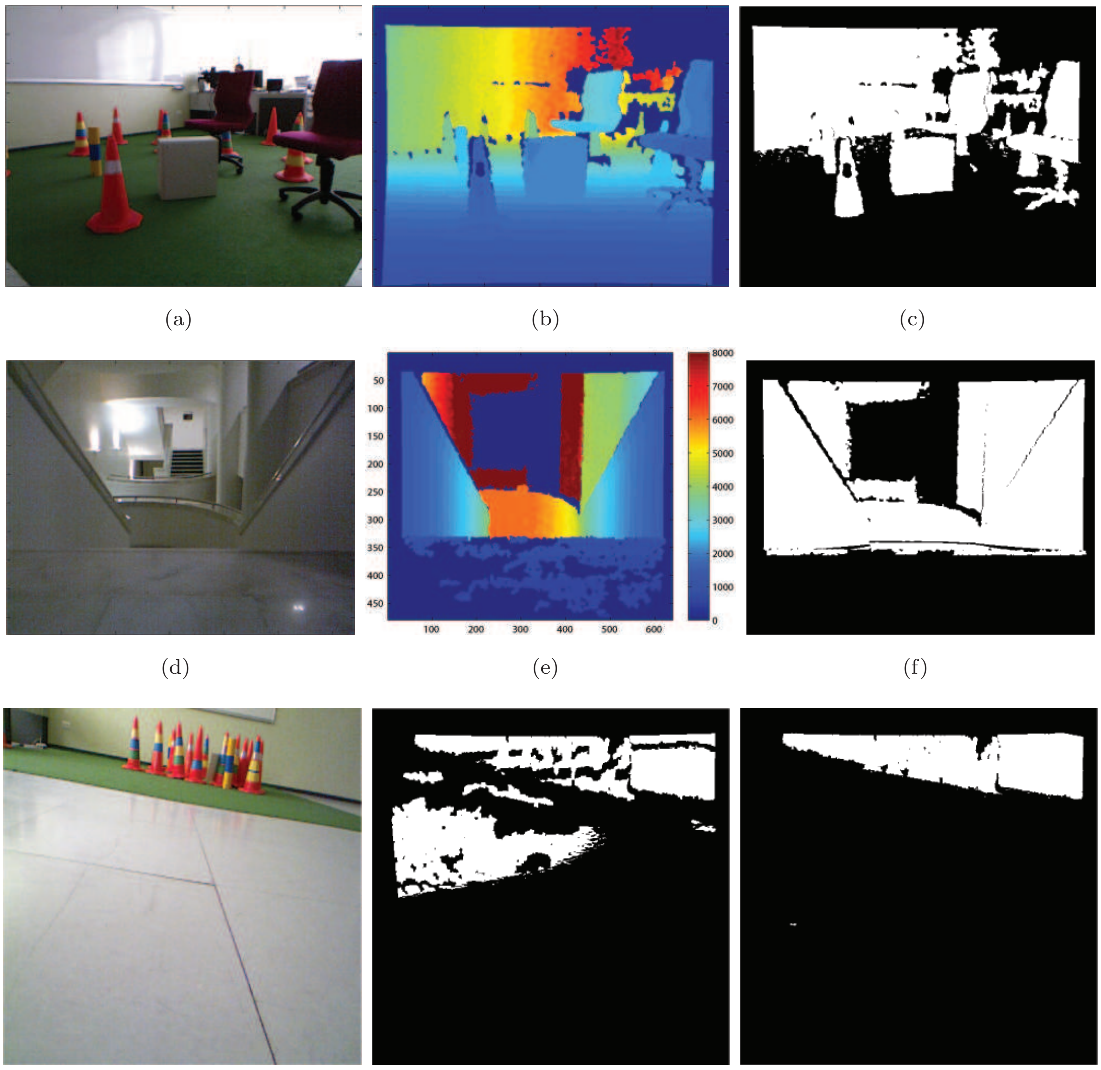
ground plane is not the largest or dominant plane. Both methods thresholds are fixed as they produce the highest respective overall accuracies in 300 frames of datasets 1 and 2. Note that V-disp marked more non-passable regions as ground plane.

If the frames are buffered beforehand and worked offline, our pitch compensated algorithm A2 processed 83 fps while running on a notebook computer with Pentium i5 480m processor using Matlab 2011a.

Additional experimental results , test frames and datasets can be found from the RAV LAB's web site[1].

---

[1] ravlab.isikun.edu.tr

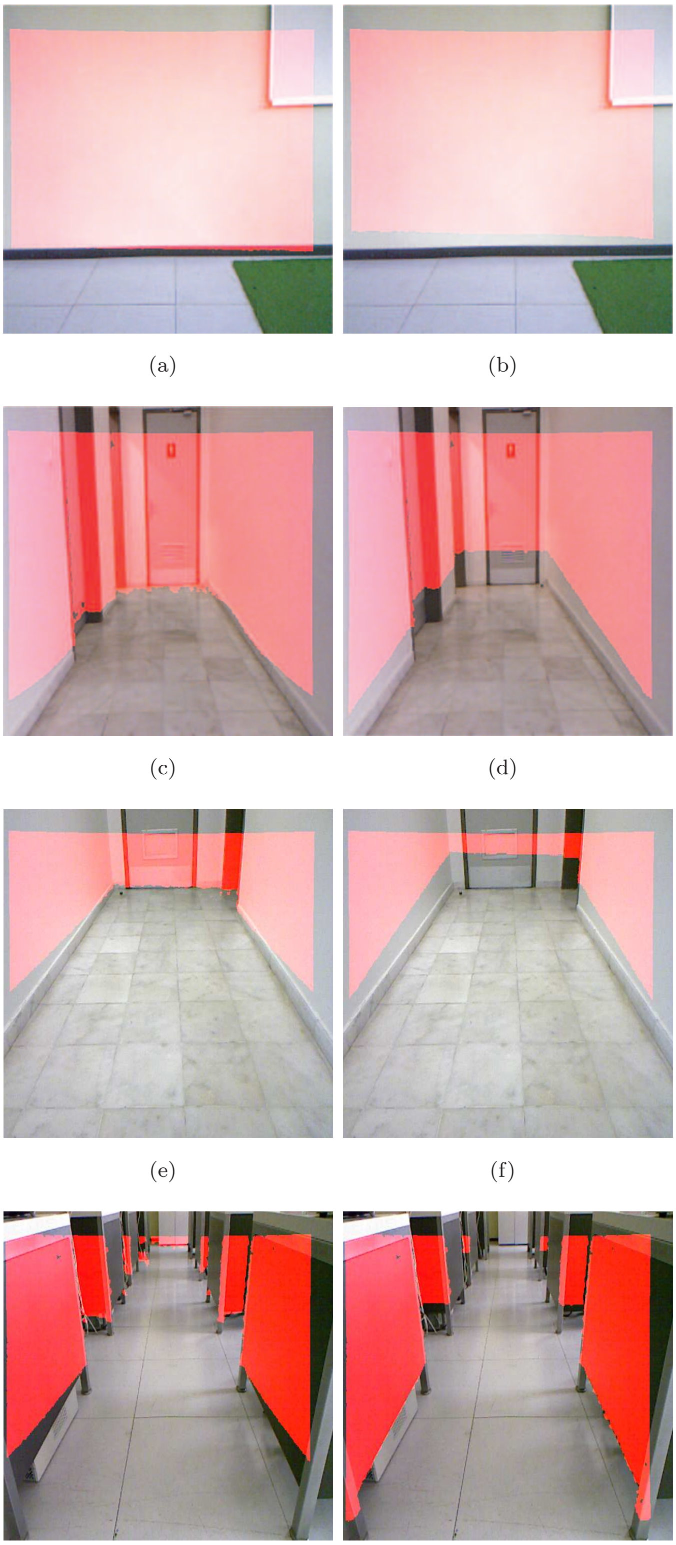


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 3.7: Experimental results from different scenes. RGB, depth-map and pitch compensated method output (white pixels represent objects whereas black pixels represent ground plane): (a),(b),(c) lab environment with many objects and reflections; (d),(e),(f) stairs; (g),(h),(i) respective outputs of pitch compensated (A-2) and pitch&roll compensated method on an image where sensor was positioned with a roll angle (A-3).



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 3.8: Comparison of pitch compensated (left) and V-disp method (right) in different scenes (red areas indicate pixels that are detected as objects): (a),(b) wall surface as dominant plane; (c),(d),(e),(f) narrow corridor; (g),(h) narrow complex passage

3.3 Discussions

In this chapter a novel, simple, and robust ground plane detection algorithm which uses depth information obtained from an RGB-D sensor is presented. Presented approach includes two different algorithms, where the first one is simple but quite robust for fixed pitch and no-roll angle scenarios, whereas the second one is more suitable for dynamic environments (e.g. moving robots or cars). Both algorithms are based on an exponential curve fit to model the ground plane which exhibits rational decreasing depth values. We compared ours to V-disp [17] method which is based on detection of a ground plane model line by Hough transform which relied on linear increasing disparity values.

It has been shown that the proposed method is better than V-disp and produces acceptable and useful ground plane-obstacle segmentations for many difficult scenes, which included many obstacles, walls, different floor surfaces, stairs, and narrow corridors.

Presented method can produce erroneous detections especially when the curve fitting is not successful. However, these situations are easy to detect by checking the RMS error of the fit which has been shown to be highly correlated with the accuracy of segmentation. Our future work will include an iterative refining procedure for curve fitting for the frames which are detected to produce high RMS fitting errors.

A point to note is about non-planar ground surfaces that few other studies in literature have devised strategies for [17, 40]. We assume here a planar ground plane model which will probably cause problems if the floor has bumps or significant inclination or declination [17]. Our future work will focus on these conditions.

Kinect has proved itself as a powerful depth sensor to be used in robotics. However, it has some limitations. Sunlight or reflective surfaces like marble floor prevent Kinect to construct a successful depth map.

Chapter 4

Landing zone detection

Automated landing zone detection is an important component to automated landing of VTOL vehicles, and there are many works which study the problem as explained in the introduction.

The work of Scherer *et al.* shows an advanced LIDAR based landing zone detection on a full-scale helicopter[48]. Point cloud data obtained from LIDAR is used to find a landing zone considering terrain/skid interaction, rotor and tail clearance, wind direction and clear approach paths. In a medium-scale platform a similar hardware configuration can be seen in [49]. LADAR sensor used to collect point cloud data and uses fuzzy logic to rank the areas to find best possible landing zone. However it is impossible to carry such an advanced sensor and its required equipment on a MAV.

One of the common approaches to detect landing zones for MAVs is to use computer vision. Sharp *et al.* [50] used a special ground mark (sign) to designate landing zones. The target sign is designed so that the segmentation could be easily performed by a standard connected components labeling algorithm ([51]) in grayscale. After the segmentation, corner detection is done over the specially designed sign to detect the ground mark. A similar approach can be seen in [52], but this time a more conventional sign for helipads, letter H is used to detect landing zones given that the helipad is easily distinguishable in the scene. These two approaches depend on the landing zone mark to be visible; requiring a low altitude that ensures mark is clearly visible.

However Cesetti *et al.* [53] suggest a more general solution by using natural landmarks, in a partially known environment. User given visual inputs of natural landmarks are matched using Scale Invariant Feature Transform (SIFT, [54]) algorithm. Other matching algorithms like Speeded Up Robust Features (SURF, [55]) , upright SURF (U-SURF [55]) and Adaptive SIFT, are also tested in their work but it is indicated that the best results are obtained using the original SIFT. To measure surface flatness, they used the variance of the optical flow, which was estimated using SIFT. These works predesignate a known safe landing zone and try to locate them when landing is commanded.

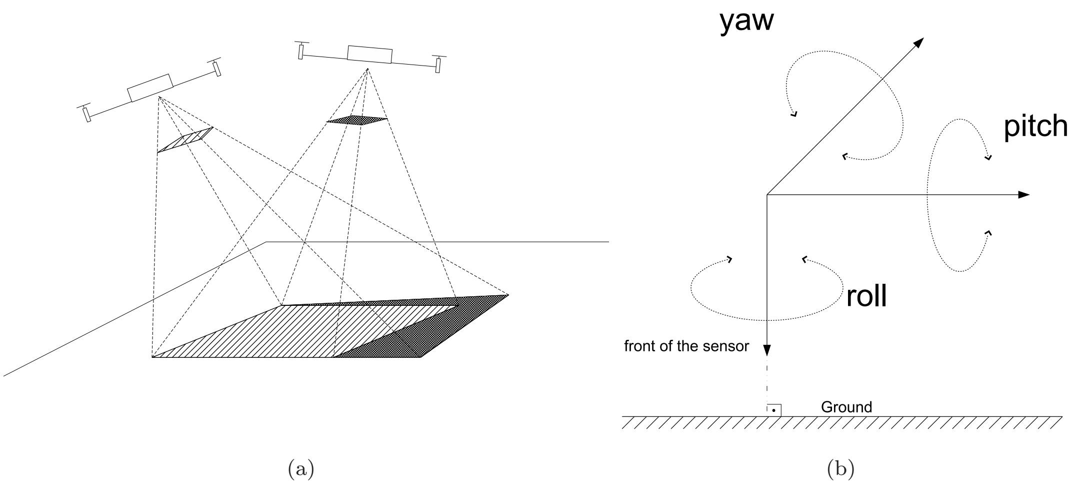
In contrast to above, in some works no safe landing zone is assumed. For example, in [56] objects or obstacles are detected by assuming that the contrast is higher at the boundaries of the obstacles than elsewhere so that the objects are distinguishable from the ground. The circular area that has a lower contrast below a threshold is chosen as landing zone. Bosch *et al.* [57] suggests a monocular image processing technique that consist of detecting planar surfaces between 2D image homographies, whereas an altimeter was used together with a single camera to construct structure from motion in [58]. Using the depth information obtained from the altimeter and two consecutive frames a digital elevation map of the scene is constructed. A point cloud is projected into the constructed structure than divided into square regions the size of the vehicle footprint. In each region a plane is fitted using least median squares (LMedS, [59]). Finally roughness and slope of the scene are examined to determine a safe landing zone.

In this chapter a new method is introduced to find a collision free landing zone using RGB-D camera. The advantage of our method is that there is no designated landing zone or mark is necessary and it can be used at night landing as the RGB-D cameras can work in pitch darkness. Also the depth is much reliable than light vision to estimate ground structure. Moreover an RGB-D camera can be easily carried by a MAV. RGB-D camera is placed underneath the vehicle, facing directly to ground. A similar installation can be seen in [60], where the RGB-D camera is used for hovering at a certain altitude. Depth map obtained from RGB-D camera is used to detect any obstacles that may interfere with the landing process.

4.1 Method

We assume that a scene that only consists of a flat ground plane is a safe landing zone. In this scene if an RGB-D camera is placed perpendicular to ground, values in the depth image D should be equal to each other which could be an ideal landing site. However, it is difficult to assume a perfect parallelity of the quad axis to ground plane. Because air vehicles change their pitch and roll angle, even when they hover. Depending on the stabilization characteristics of the vehicle, these movements complicate quad's viewing angle and make difficult to interpret the ground structure (Figure 4.1(a)). Although our quad has gyroscopic sensors to stabilize the roll and pitch, additional stabilization hardware can be used to minimize affects to the camera axis. Nevertheless, if an image of the scene is acquired with a non-zero pitch angle to the ground, the linear distance from the sensor will not be projected to the depth map linearly, but exponentially. This phenomena and its solution was discussed in detail in Chapter 3. Hence this allows us to use roll and pitch compensating ground plane detection method explained in Section 3.1.2 with an additional preprocess.

Axis movements have different affects on the depth map, due to the change in the projection of the depth map. Roll axis movements of the sensor only rotates the image but has no longer any effect on the depth map values. However yaw axis movements (Figure 4.1(b)) can change depth map values as they change the projected scene. It should be taken into consideration that the mentioned axes are for the camera movements but not for the vehicle movements. Method presented in Chapter 3 works on the principle that the ground plane curve is a constant or



(a) (b)

Figure 4.1: (a) Viewing perception from quad; (b) Axes of the camera

increasing function from bottom of the image to the top. Because of that there are three different important conditions, which must be taken into account.

Change in the roll axis leads to a crosswise increasing ground plane curve. However the method introduced in Section 3.1.2, searches for a lengthwise ground plane curve. To solve this problem the depth image must be rotated from the center so that every horizontal line will have close depth values and the values increase vertically. Roll compensation method presented in Section 3.1.2 can be used here, however due to the fact that yaw angle with zero pitch causes crosswise ground plane curve, the rotation search angle range must be increased to$-90°$ to $90°$.

Any positive pitch axis movement changes the ground plane curve from planar to an decreasing rational function. So positive change in the pitch axis does not require any additional process before using our ground plane detection method.

But if there is a negative change in the pitch axis, ground plane curve becomes an increasing function (Figure 4.2) . Without any changes the previously mentioned method can not find a ground plane curve as it relies on an decreasing function. Hopefully a simple approach as flipping the image upside down, change the orientation of the ground plane curve and ensures that it is an decreasing function. In order to check if a flipping is required, derivative of the depth envelope is used.

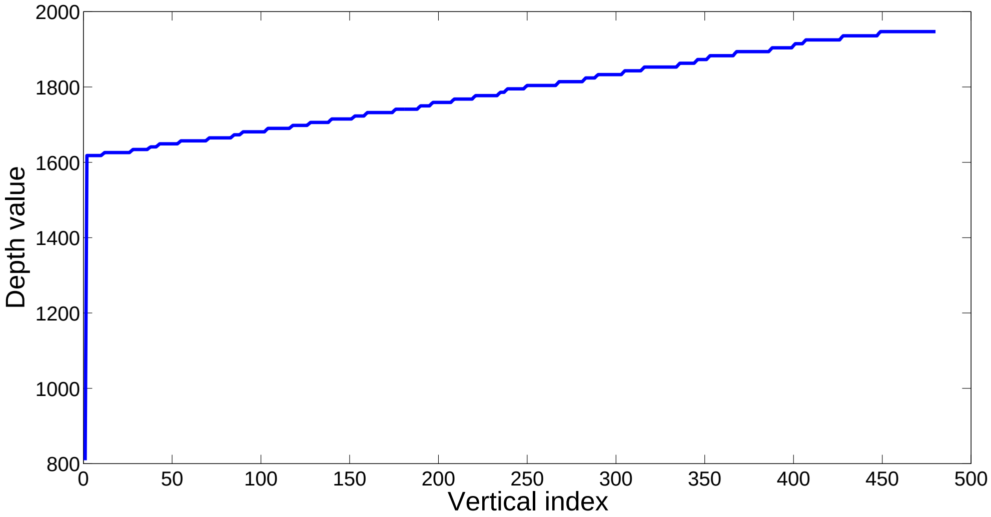


Figure 4.2: Depth envelope (E) for negative pitch angle

Let, (E) denote depth envelope, (C) number of columns, $d(E)/dx$ must be a positive function if the ground plane is an increasing function. If it is found to be an increasing function, the image is flipped upside down.

$$S = \sum_{x=1}^{C} \frac{dE}{dx} \tag{4.1}$$

S value shows total derivative of the depth envelope (E). By taking these 3 conditions into consideration, any outcome from combination of these conditions have been solved.

Acquired image is rotated until equal or very similar depth values along every horizontal line (i.e. rows) is acquired. Then the image is flipped if S>0 then A2 method given in Section 3.2 is applied.

4.2 Experiments

We captured different ground scenes about 2m height, 11 from our lab environment, 40 from outside including sunlight illuminated scenes and night scenes. Scenes includes possible landing zones and non-landable zones with flat surface objects like desks, chairs and other kind of objects like cones, garden lightings and small trees. Outdoor scenes captured at different times of the day to see the affect of sunlight. It was known that it is not possible to get a depth map reading from a scene that directly illuminated by the sun light. However as it is a hardware limitation such scenes with no usable depth readings are excluded from experiments. While capturing the scenes no information about the system's axes or the sensor's axes has been recorded. Success of the algorithm is tested for the objects that are at least 5cm wide. Response to smaller objects like spikes in the ground is not tested due to occurrence of the situation.

The results can be seen from Figure(4.3-4.12). For each scene RGB image, depth image, rotation corrected RGB, rotation corrected depth image and method output is shown. Rotation correction is done automatically by the given method. Black pixels represents ground plane and white pixels represents objects as used before. Figure 4.3 shows hovering position at the edge of a desk. Desk is successfully detected. Figure 4.4 shows a cone as a small obstacle which prevent safe landing. Figure 4.5 shows a crowded scene. Figure 4.6 only consists of the ground plane. Boundaries of objects can be seen at the edges of the RGB image however depth readings are not available for these parts.

To test as many different scenes as possible Figure(4.7,4.8) shows a scene from the ceiling which includes a row covering object. Ventilation pipe, fire alarm and lighting can be seen as obstacles. Ventilation pipe covers whole row at the scene. In Figure 4.8 it can be seen that there are some error at the top of the rotated image, which is caused by the row covering object at the bottom of the rotated image. However the method indicates this is not a valid landing zone.

Outdoor environment test result are shown in Figure(4.9-4.12). For scenes captured during daytime, it must be noted that there is no direct sunlight. Figure 4.10 and Figure 4.9 shows the same spot during daytime and night. Night scenes (Figure 4.10, 4.11) shows the big advantage of the RGB-D cameras, regular

RGB cameras inadequate at capturing images at low light or no light conditions while RGB-D cameras are not affected at all.

4.3 Discussions

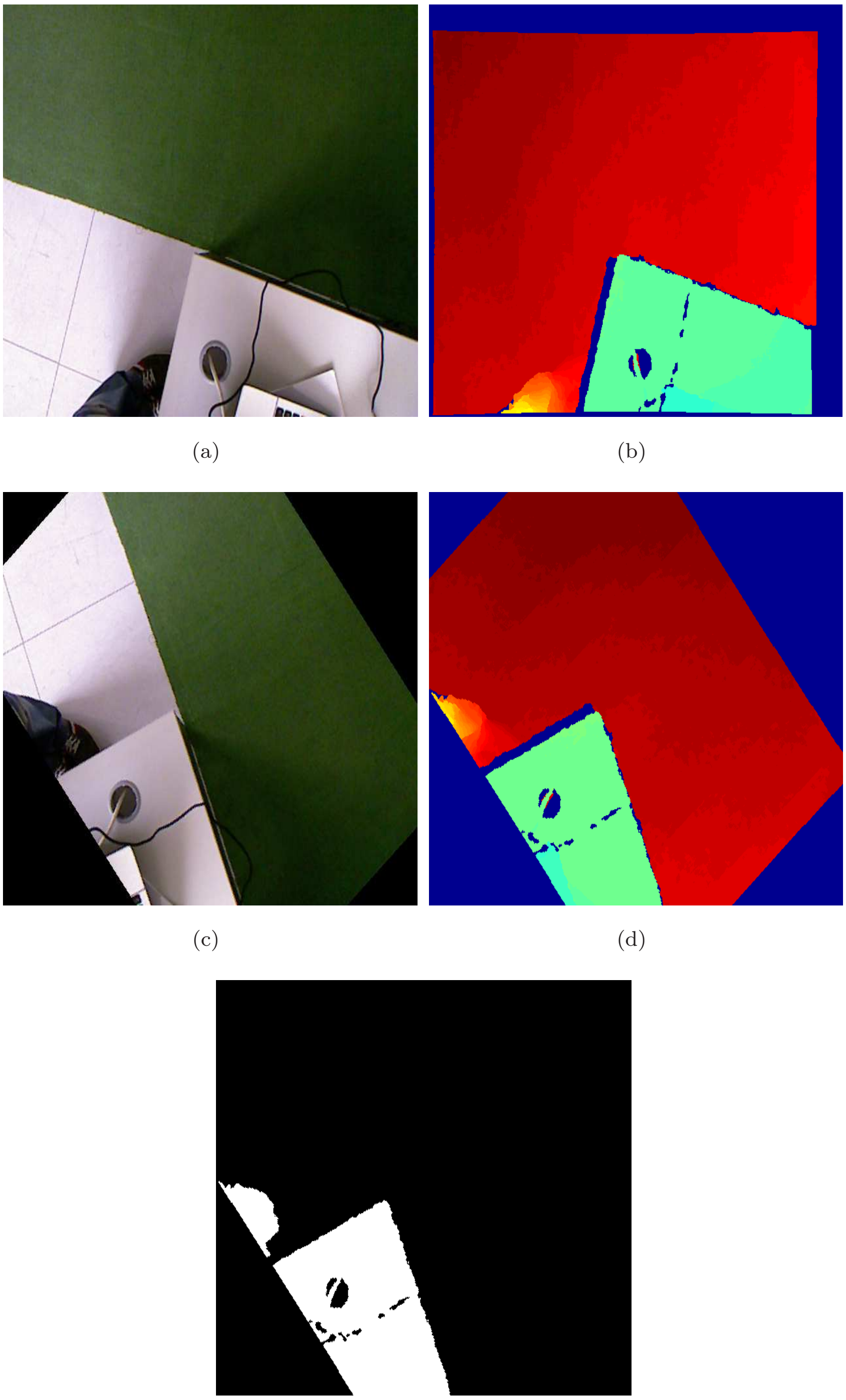
In this chapter a novel and simple safe landing zone detection algorithm which uses only the depth information obtained from an RGB-D sensor is presented. Presented approach consists of a preprocessing stage and our ground plane detection algorithm (Chapter 3).

Initial experiments have shown that the proposed method is successful in detecting ground plane obstacles. Pitch and yaw axis movements of the camera is successfully compensated with the preprocess and objects are successfully detected using ground plane detection algorithm.

Only drawback of this technique is that if there is a whole row covering object at the bottom of the processed depth image, the method is inadequate at finding the object position (Figure 4.8). But the outcome gives the information that there exists an object at landing zone, indicating an inappropriate landing zone.

In the preprocess stage, turning the depth image to compensate the yaw angle method is computationally costly. However yaw and pitch angle is already known for the quadcopter. If these values are obtained from the IMU, the algorithm will run much more faster. Although synchronizing the IMU data and RGB-D data will be an issue in that case as both devices give the data when it is available on request.

Slope of the surface (inclination or declination) is not detectable with the proposed algorithm and so it is not taken into account. If the data obtained from the IMU is used with the current state of the algorithm, slope of the surfaces might be detected. Then the maximum level of inclination can be determined for safe landing zone. It would also advance the speed of the method as the axis movements can be neutralized in one step with the IMU data. Use of the depth and

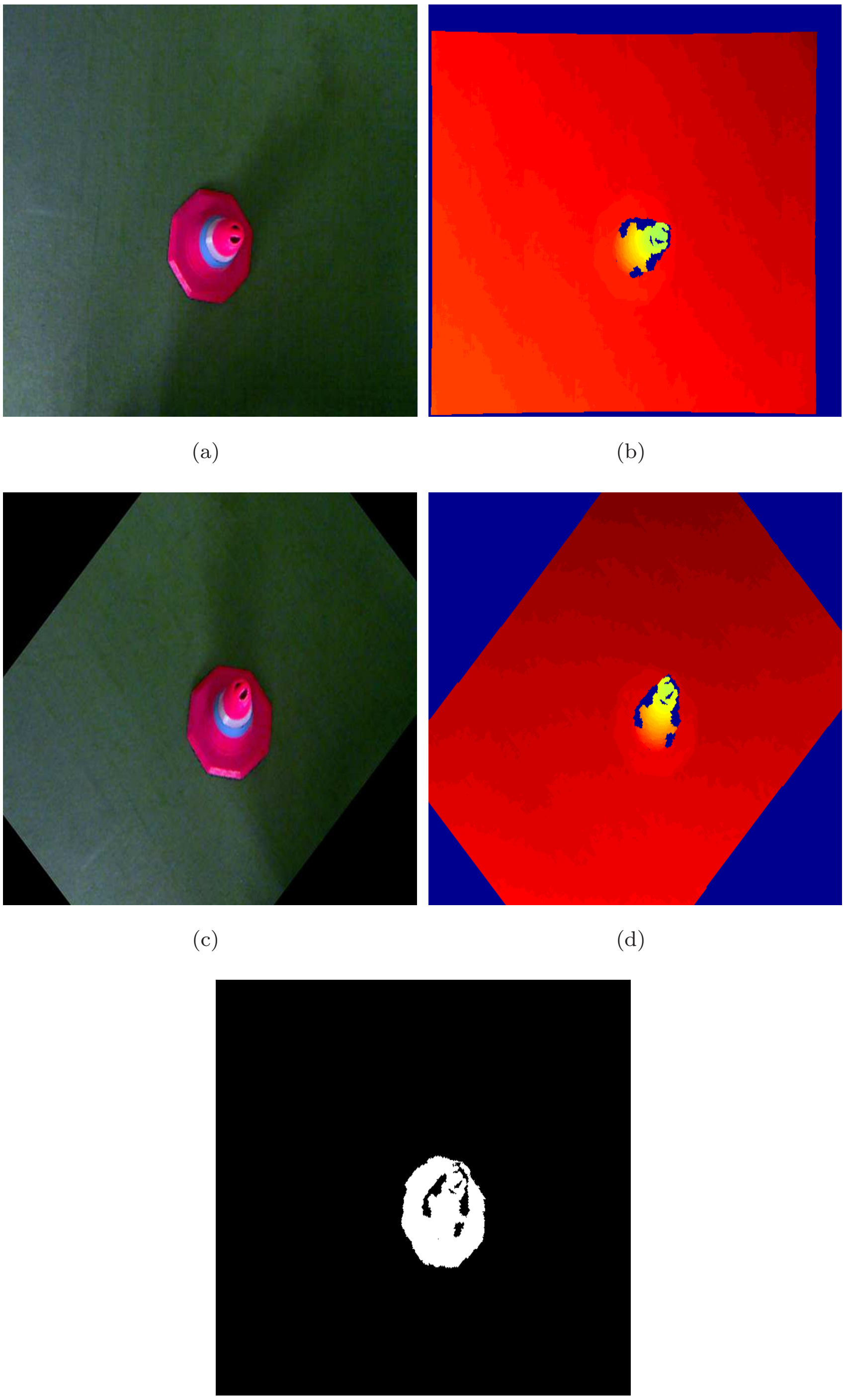


(a) (b)

(c) (d)

(e)

Figure 4.3: Scene with partial desk and ground plane. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane



(a)

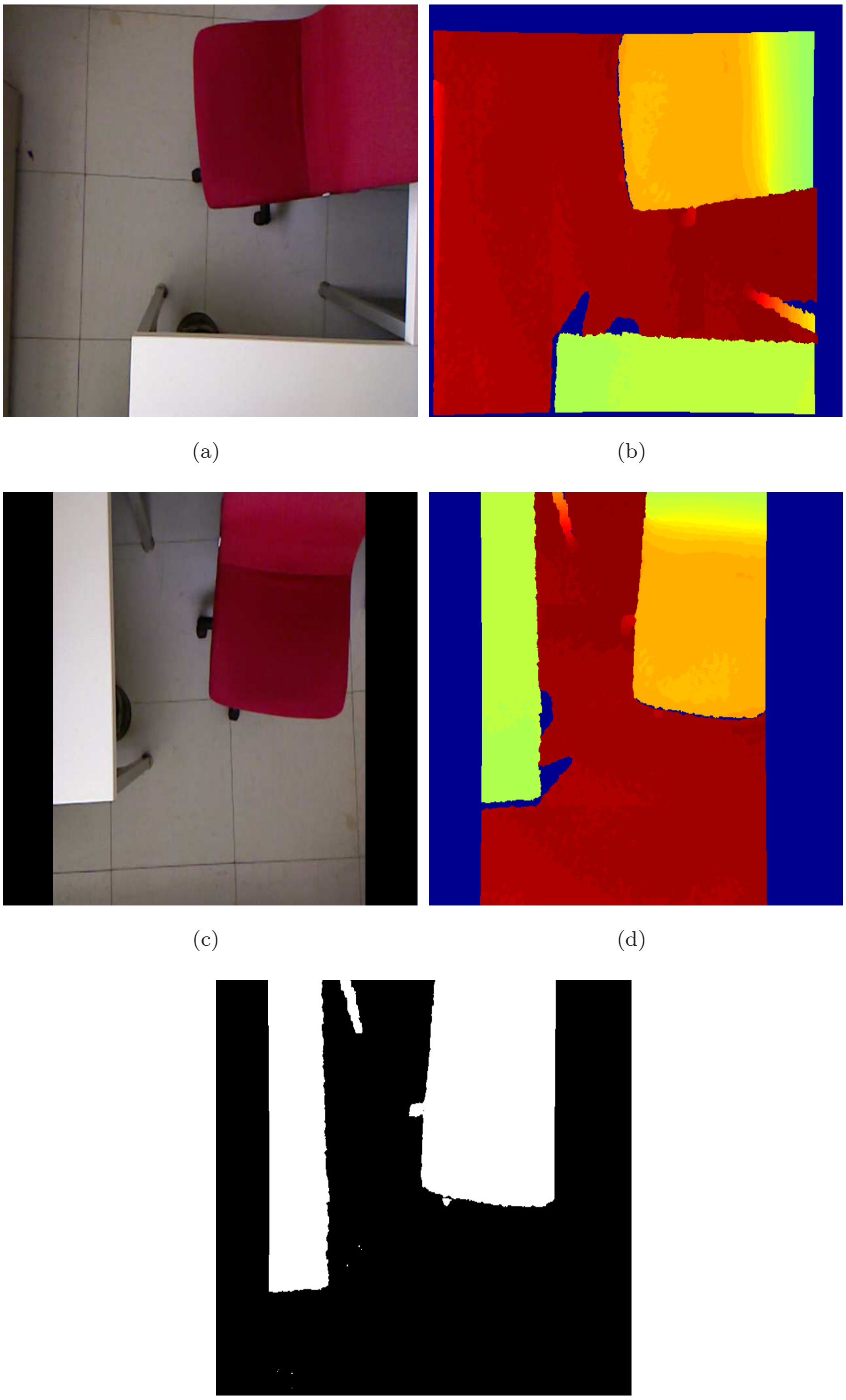
(b)

(c)

(d)

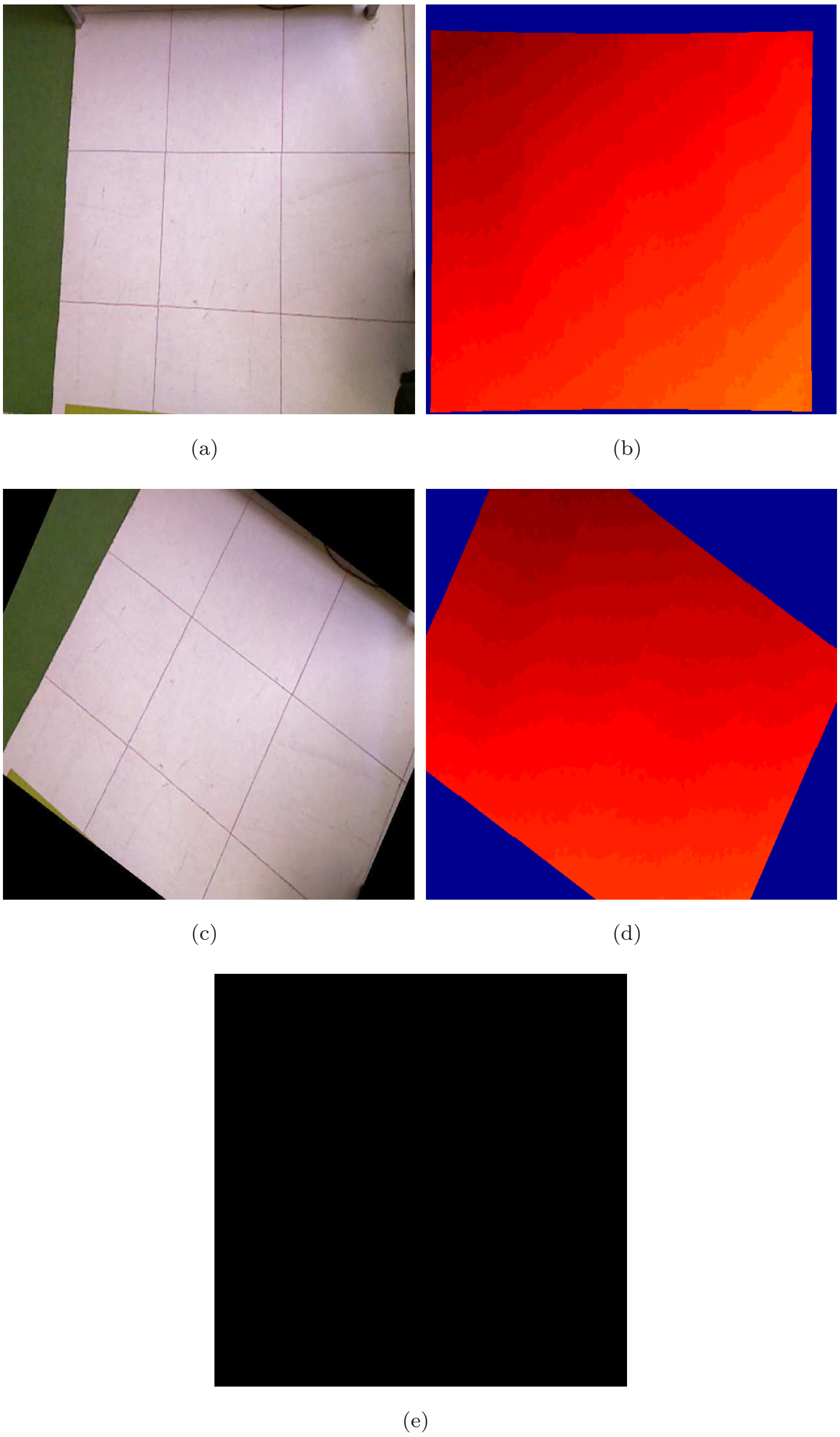
(e)

Figure 4.4: Scene with a cone and ground plane. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane



(a) (b) (c) (d) (e)

Figure 4.5: Scene with desk,chair and ground plane. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane

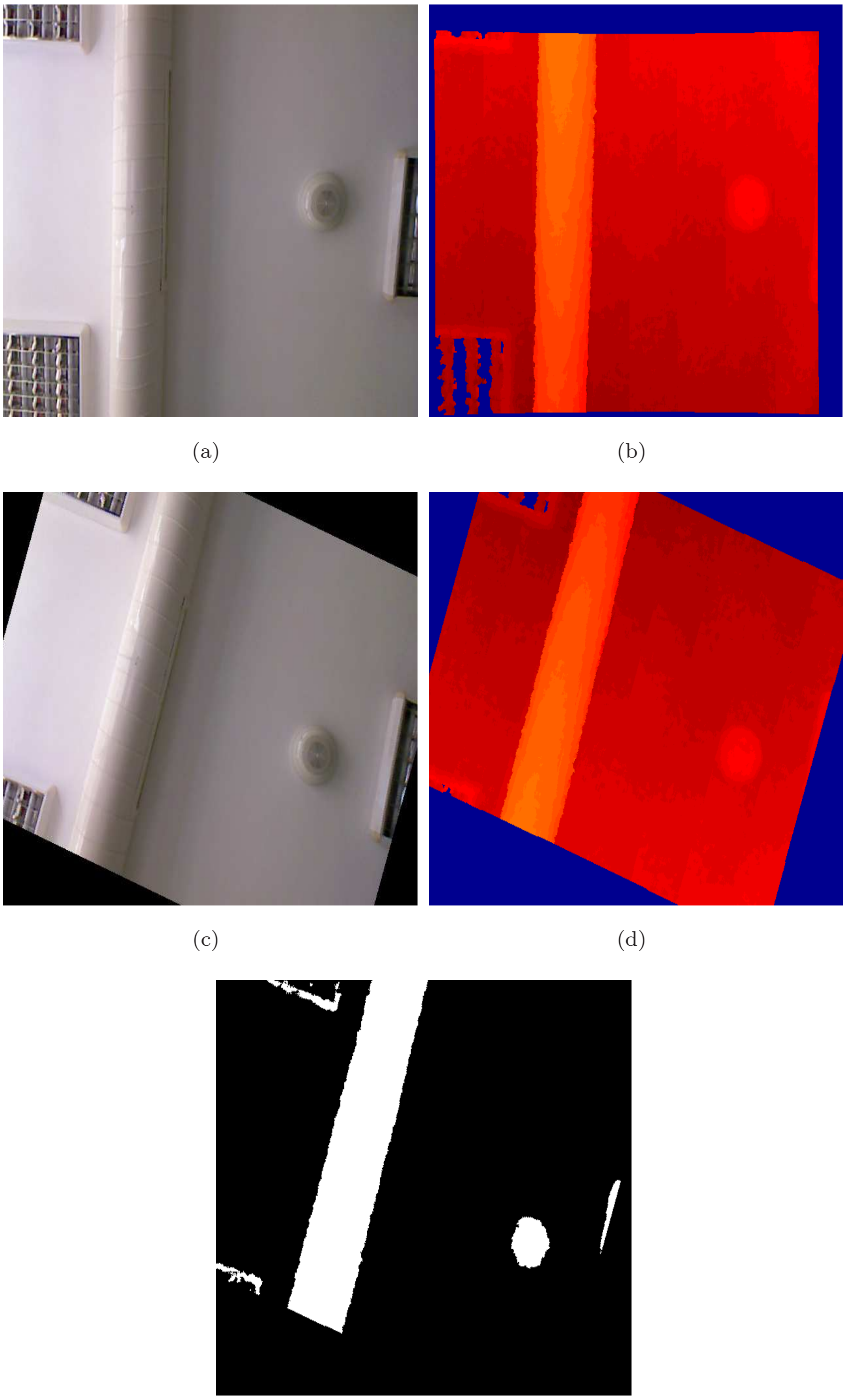


(a) (b)

(c) (d)

(e)

Figure 4.6: Scene with only ground plane, possible landing zone. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane



(a)

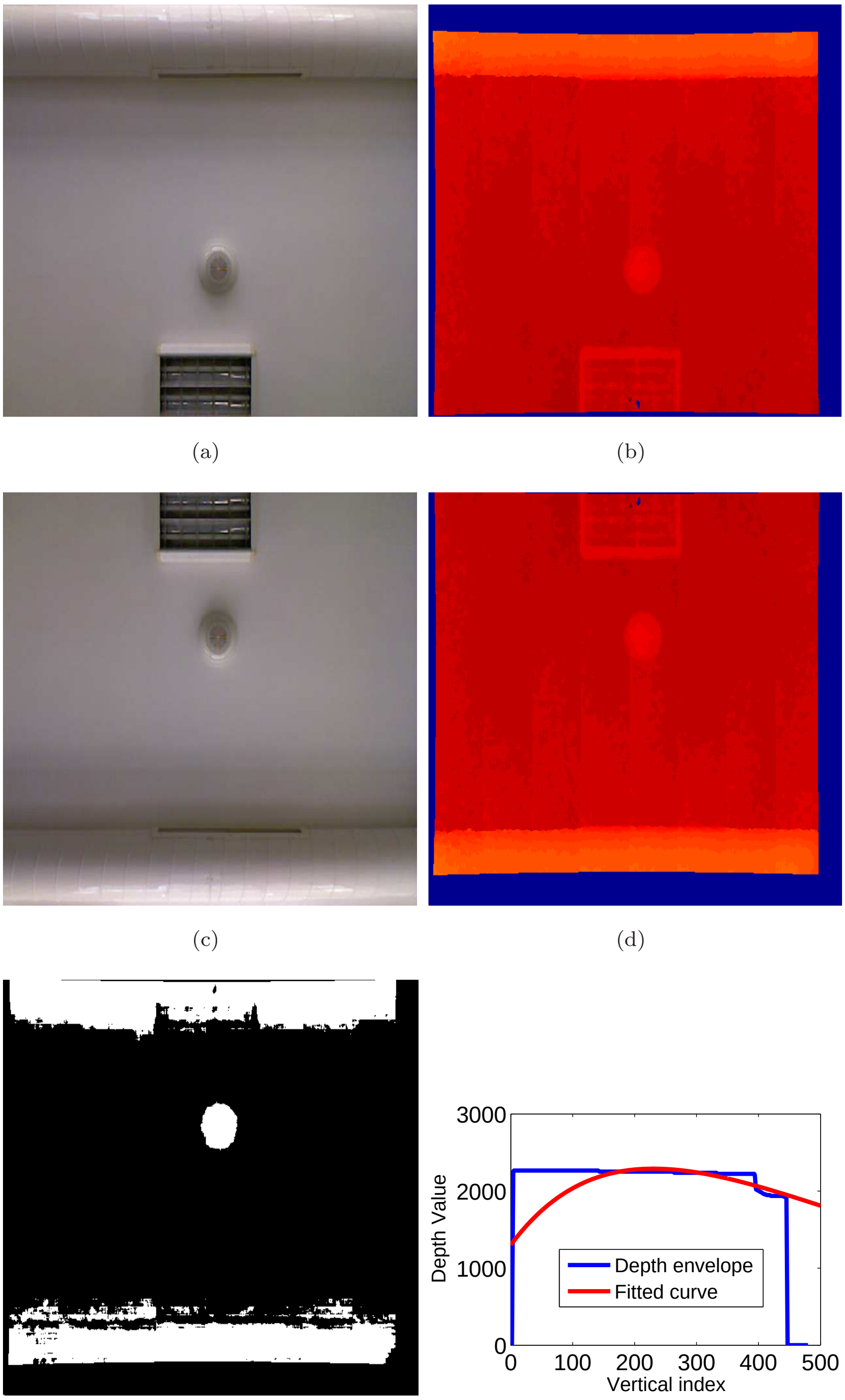
(b)

(c)

(d)

(e)

Figure 4.7: Ceiling with obstacles (tested for row covering object). (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane

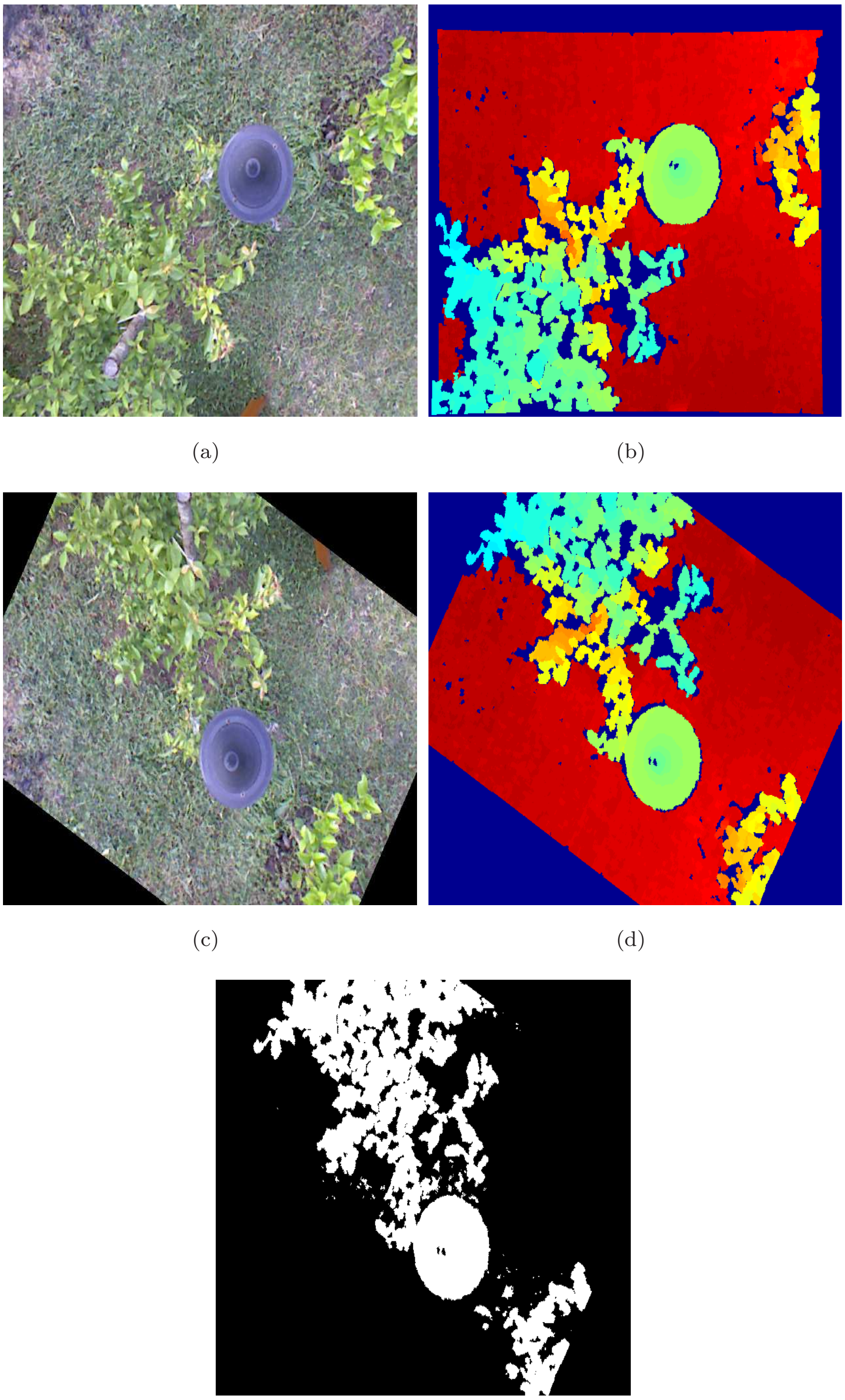


(a) (b)

(c) (d)

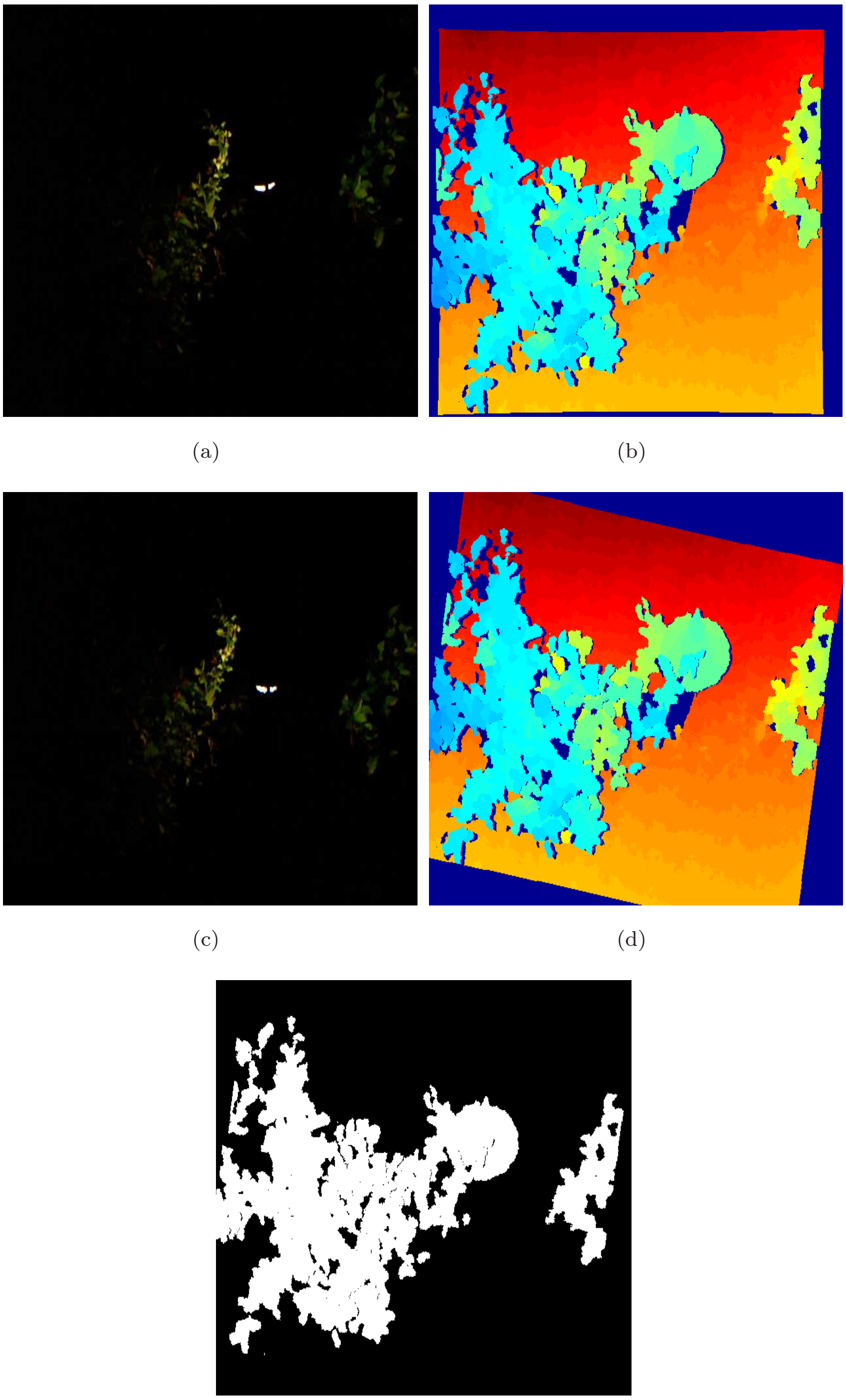
(e) (f)

Figure 4.8: Ceiling with obstacles (tested for row covering object). (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane, (f) depth envelope of the scene and erroneous fitted curve



(a) (b) (c) (d) (e)

Figure 4.9: Outdoor environment. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane

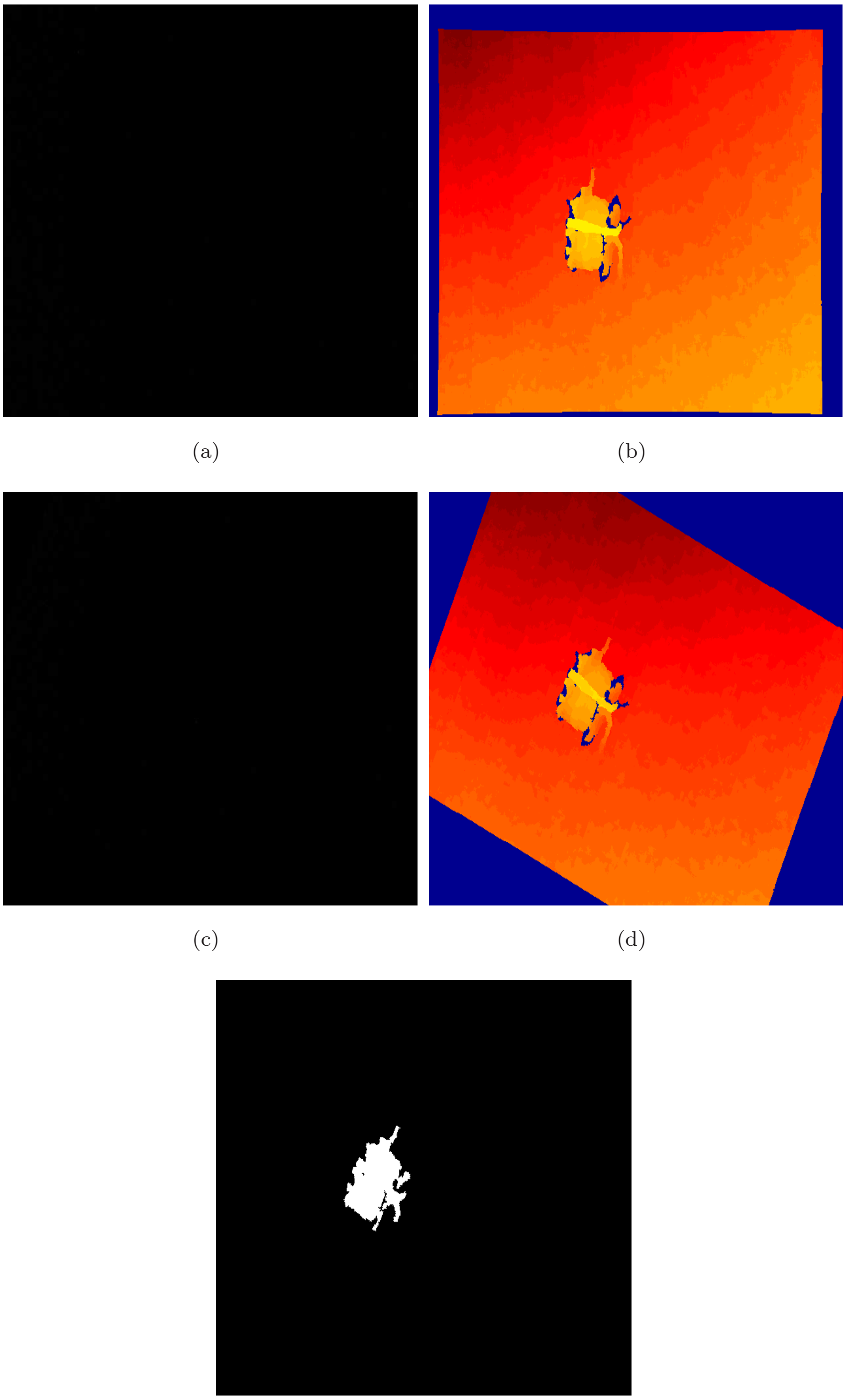


(a) (b)

(c) (d)

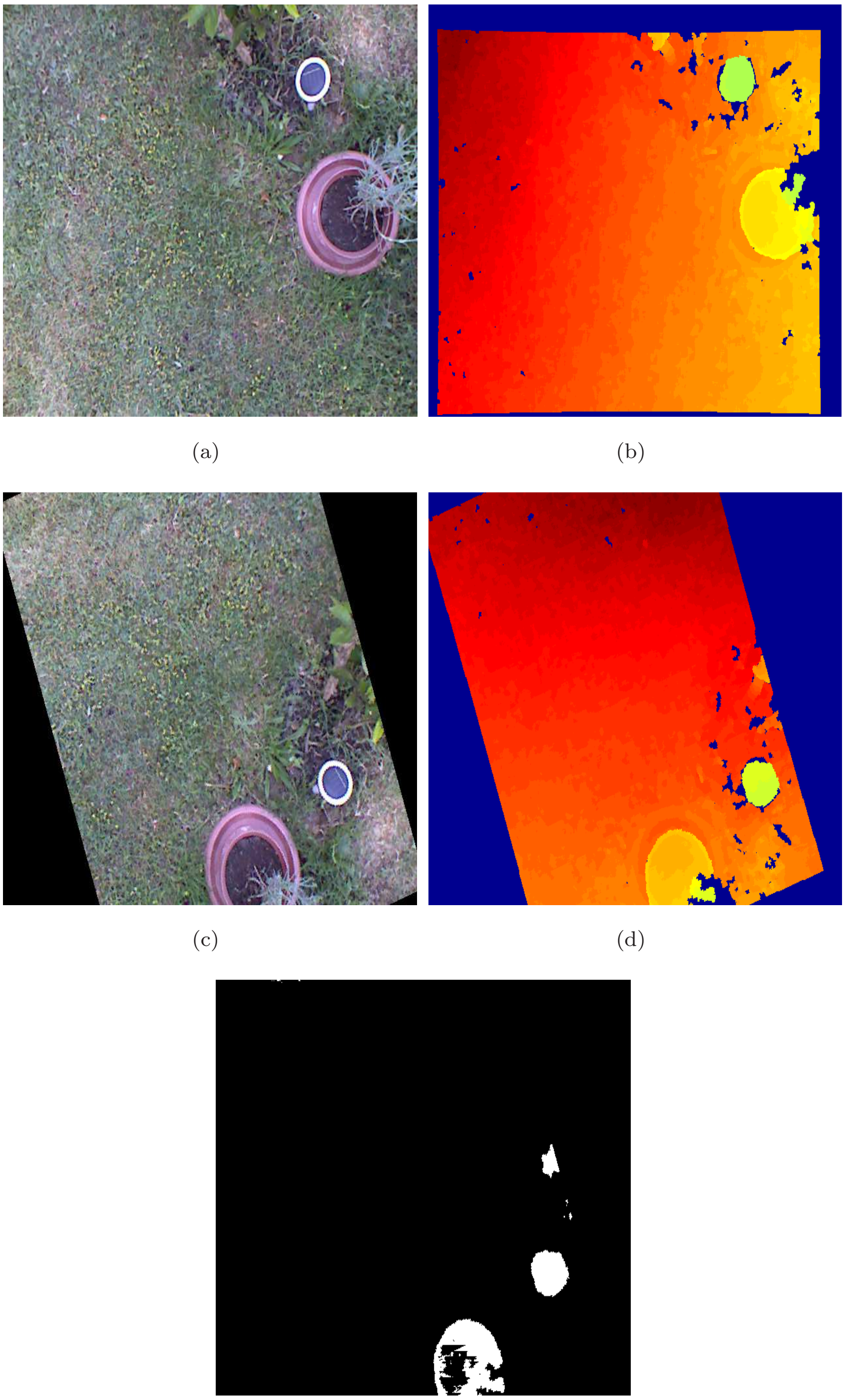
(e)

Figure 4.10: Outdoor environment of Figure 4.9 at night. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane



(a) (b) (c) (d) (e)

Figure 4.11: Outdoor environment at night, as there is no light source RGB image is shown black. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane



(a) (b) (c) (d) (e)

Figure 4.12: Outdoor environment. (a) RGB image, (b) depth-map, (c) rotation corrected RGB image, (d) rotation corrected depth-map; (e) method output, white pixels represent objects whereas black pixels represent ground plane

IMU data would be enough for obstacle detection on planar surfaces. However to detect obstacles on sloped surfaces, our method would be also required.

Since the method locates, detects obstacles, it is possible to categorize the scenes as not-safe/safe landing zones by simply obstacle presence. However, in cluttered environments and outside scenes, it must be crucial to rate how safe or how non-safe is that landing area. This can be performed by taking into account the ratio of obstacles to the non-obstacle regions as well as quad's height, obstacle height, ground flatness etc. These issues must be addressed in the future.

Conclusion

The aim of this thesis is to build an autonomous quadcopter which is capable of automated landing. To achieve this goal there are two essential tasks: equipping a flyable quadcopter platform with several sensors; detection of the ground plane and potential obstacles for safe landing area detection.

In this study we acquired two different commercially available quadcopter frames and a Ardu Pilot Mega control card to build a low cost, easy to implement, and stable platform. An open source firmware, named Arducopter[16], is used for the control card. This framework has the abilities of take-off, landing, hovering, and point to point navigation *on command* and performing a pre-programmed flight path navigation. We modified the acquired frame to integrate a sonar sensor, a small size pc (FitPC[24]), and a Microsoft Kinect[25] RGB-Depth camera. Additionally, we installed a CCD-camera attached to a two-axis gimbal and a downlink (radio) system to transmit real time video to ground station, as useful payloads.

After building the quadcopter platform and integrating the necessary equipment, we focused on ground plane detection problem, as it is an essential task for automated navigation and landing. We introduced a novel, simple, and robust ground plane detection method[61] which uses depth information obtained from an RGB-D camera ,and presented it in *21st Signal Processing and Communications Applications Conference* in Girne, Turkish Republic of Northern Cyprus. It has been shown that the method is able to compensate for changing pitch and roll view angles. Moreover, our method was compared with an existing method from the literature, i.e. V-disparity algorithm [17]. It has been shown that our algorithm is better than V-disparity method and produces useful ground plane-obstacle segmentations for many different cases. This algorithm is generic in the sense that it can be used for different forward facing RGB-D placements, for example in ground vehicles or robots.

It has been shown that our generic ground plane detection method is inadequate for landing zone detection and had to be improved to adjust to down-facing view projections of the quadcopter. To detect possible landing zones for quadcopter landing, a novel and simple safe landing zone detection method is introduced. The method combines a preprocessing stage and our ground plane detection algorithm. The preprocessing stage successfully compensated the angular change in the camera's down-facing position and our ground plane detection algorithm successfully segmented the ground plane and obstacles in the scene. It is also shown that the use of RGB-D camera allows the ground plane detection and landing zone detection even in no-light conditions, thanks infra-red projection and sensing. However in scenes that are directly illuminated by the sunlight RGB-D does not work due to infra-red interference from the sun.

The proposed methods are very accurate and applicable to most aerial vehicles (MAV) since that the payload weight is very convenient for small aircrafts and also that the processing power needed for the methods is achievable onboard. The usability of the methods in very low light conditions, such as the night time or even in the pitch dark is another advantage.

The ground plane detection method can produce erroneous detections; however our method has the ability to detect these situations. To handle these errors, it may be possible to develop an iterative refining procedure for curve fitting, which may improve the consistency of the method further.

While detecting the safe landing zones, the algorithm ignores height information and does not consider surface inclination or declination. It is possible to use quad's inertial state information, with RGB-D data (including height) for better classification of zones. Finally, as we could not implement the RAVLAB Control Station completely, currently; we cannot command the quadcopter to land, even though we detect safe landing zones. In the future, we plan to complete the integration to achieve a fully autonomous landing.

References

[1] Anka - tusaş-türk havacılık ve uzay sanayii a.ş. `https://www.tai.com.tr/tr/proje/anka`. Accessed: 2013-05-15.

[2] Global hawk. `http://www.northropgrumman.com/Capabilities/GlobalHawk/Pages/default.aspx`. Accessed: 2013-05-15.

[3] Yamaha rmax — the high performance unmanned helicopter designed for a wide range of industrial uses. `http://rmax.yamaha-motor.com.au/`. Accessed: 2013-05-15.

[4] Pixhawk research project - eth pixhawk: Mav computer vision. `https://pixhawk.ethz.ch/`. Accessed: 2013-05-15.

[5] Lockheed martin & middot; high altitude airship. `http://www.lockheedmartin.com/us/products/lighter-than-air-vehicles/haa.html`. Accessed: 2013-05-15.

[6] Nano air vehicle (nav) - aerovironment, inc. `http://www.avinc.com/nano/`. Accessed: 2013-05-15.

[7] Predator uas. `http://www.ga-asi.com/products/aircraft/predator.php`. Accessed: 2013-05-20.

[8] Introduction - schiebel. `http://www.schiebel.net/Products/Unmanned-Air-Systems/CAMCOPTER-S-100/Introduction.aspx`. Accessed: 2013-05-20.

[9] Fire scout. `http://www.northropgrumman.com/Capabilities/FireScout/Pages/default.aspx`. Accessed: 2013-05-20.

[10] Boeing: Unmanned little bird. `http://www.boeing.com/boeing/rotorcraft/military/ulb/`. Accessed: 2013-05-20.

[11] Sung-Yeol Kim, Eun-Kyung Lee, and Yo-Sung Ho. Generation of roi enhanced depth maps using stereoscopic cameras and a depth camera. *Broadcasting, IEEE Transactions on*, 54(4):732–740, Dec.

[12] N. Özalp, Ö.K. Şahingöz, and Uğur Ayan. Autonomous unmanned aerial vehicle route planning. In *21st Signal Processing and Communications Applications Conference (SIU)*, 2013.

[13] H. Ergezer and M.K. Leblebicioğlu. 3d path planning for unmanned aerial vehicles. In *21st Signal Processing and Communications Applications Conference (SIU)*, 2013.

[14] Jiajia Shang and Zhongke Shi. Vision-based runway recognition for uav autonomous landing. *International Journal of Computer Science and Network Security*, 7(3):112–117, 2007.

[15] Farid Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[16] Arducopter - arduino-based autopilot for mulitrotor craft, from quadcopters to traditional helis - google project hosting. `http://code.google.com/p/arducopter/`. Accessed: 2013-05-01.

[17] R. Labayrade, D. Aubert, and J. P Tarel. Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 646–651 vol.2, June.

[18] Kenzo Nonami, Farid Kendoul, Satoshi Suzuki, Wei Wang, and Daisuke Nakazawa. *Autonomous Flying Robots: Unmanned Aerial Vehicles and Micro Aerial Vehicles*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[19] A. Tayebi and S. McGilvray. Attitude stabilization of a vtol quadrotor aircraft. *Control Systems Technology, IEEE Transactions on*, 14(3):562–571, 2006.

[20] Jdrones, r/c uavs and more.... `http://store.jdrones.com/default.asp`. Accessed: 2013-05-20.

[21] Radio control planes, helicopters, cars, boats, fpv and quadcopters - hobbyking. `http://www.hobbyking.com/hobbyking/store/index.asp`. Accessed: 2013-05-20.

[22] Lithium iron phosphate battery technology - a123 systems. `http://www.a123systems.com/lithium-iron-phosphate-battery.htm`. Accessed: 2013-05-20.

[23] M. Margolis. *Arduino Cookbook, 2nd Edition*. O'Reilly Media, 2011.

[24] Fit-pc2/i - fit-pc. `http://www.fit-pc.com/web/fit-pc/fit-pc2-i/`. Accessed: 2013-05-20.

[25] Kinect for windows - voice, movement & gesture recognition technology. `http://www.microsoft.com/en-us/kinectforwindows/`. Accessed: 2013-05-20.

[26] J. Stowers, M. Hayes, and A. Bainbridge-Smith. Altitude control of a quadrotor helicopter using depth map from microsoft kinect sensor. In *Mechatronics (ICM), 2011 IEEE International Conference on*, pages 358–362, April.

[27] Caroline Rougier, Edouard Auvinet, Jacqueline Rousseau, Max Mignotte, and Jean Meunier. Fall detection from depth map video sequences. In *Proceedings of the 9th international conference on Toward useful services for elderly and people with disabilities: smart homes and health telematics*, ICOST'11, pages 121–128, Berlin, Heidelberg, 2011. Springer-Verlag.

[28] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2), 2012.

[29] Mission - ardupilot-mega - official arduplane repository - google project hosting. `http://code.google.com/p/ardupilot-mega/wiki/Mission`. Accessed: 2013-05-20.

[30] Mavlink micro air vehicle communication protocol - qgroundcontrol gcs. `http://qgroundcontrol.org/mavlink/start`. Accessed: 2013-05-01.

[31] ardupilot-mega - official arduplane repository - google project hosting. `http://code.google.com/p/ardupilot-mega/`. Accessed: 2013-05-20.

[32] Px4fmu autopilot / flight management unit. `https://pixhawk.ethz.ch/px4/modules/px4fmu`. Accessed: 2013-05-20.

[33] Intro. `http://slugsuav.soe.ucsc.edu/index.html`. Accessed: 2013-05-20.

[34] Home - qgroundcontrol gcs. `http://www.qgroundcontrol.org/`. Accessed: 2013-05-20.

[35] happykillmore-gcs - happykillmore's ground control station - google project hosting. `http://code.google.com/p/happykillmore-gcs/`. Accessed: 2013-05-20.

[36] Documentation - ros wiki. `http://www.ros.org/wiki/`. Accessed: 2013-05-20.

[37] Ros (robot operating system) mavlink integration tutorial - qgroundcontrol gcs. `http://qgroundcontrol.org/dev/mavlink_ros_robot_operation_system_integration_tutorial`. Accessed: 2013-05-20.

[38] F Li, J M Brady, I Reid, and H Hu. Parallel image processing for object tracking using disparity information. In *In Second Asian Conference on Computer Vision ACCV '95*, pages 762–766, 1995.

[39] Stephen Se and Michael Brady. Ground plane estimation, error analysis and applications. *Robotics and Autonomous Systems*, 39(2):59 – 71, 2002.

[40] Qian Yu, Helder Araújo, and Hong Wang. A stereovision method for obstacle detection and tracking in non-flat urban environments. *Auton. Robots*, 19(2):141–157, September 2005.

[41] Camillo J. Taylor and Anthony Cowley. Parsing indoor scenes using rgb-d imagery. In *Robotics: Science and Systems*, July 2012.

[42] K. Gong and R. Green. Ground-plane detection using stereo depth values for wheelchair guidance. In *Image and Vision Computing New Zealand, 2009. IVCNZ '09.*, pages 97–101, Nov.

[43] C. Zheng and R. Green. Feature recognition and obstacle detection for drive assistance in indoor environments. In *Image and Vision Computing New Zealand, 2011. IVCNZ '11. 26th International Conference*, pages 58–63, Nov.

[44] Dirk Holz, Stefan Holzer, Radu Bogdan Rusu, and Sven Behnke. Real-Time Plane Segmentation using RGB-D Cameras. In *Proceedings of the 15th RoboCup International Symposium*, volume 7416, pages 307–317, Istanbul, Turkey, July 2011. Springer.

[45] Can Erdogan, Manohar Paluri, and Frank Dellaert. Planar segmentation of rgbd images using fast linear fitting and markov chain monte carlo. In *CRV'12*, pages 32–39, 2012.

[46] Luke Wang, Russel Vanderhout, and Tim Shi. Computer vision detection of negative obstacles with the microsoft kinect. *University of British Columbia. Engineering Projects Project Lab. ENPH 459, Project Conclusion Reports*, 2012.

[47] İ.K. İyidir, F.B. Tek, and D. Kırcalı. Adaptive visual obstacle detection for mobile robots using monocular camera and ultrasonic sensor. In *Proceedings of ECCV2012 Ws/Demos, Part II, LNCS 7584*, pages 526–535, Florence, Italy, October 2012.

[48] Sebastian Scherer, Lyle Chamberlain, and Sanjiv Singh. Autonomous landing at unprepared sites by a full-scale helicopter. *Robot. Auton. Syst.*, 60(12):1545–1562, December 2012.

[49] M. D. Takahashi, A. Abershitz, R. Rubinets, and M. Whalley. Evaluation of Safe Landing Area Determination Algorithms for Autonomous Rotorcraft Using Site Benchmarking. *American Helicopter Society 67th Annual Forum*, 2011.

[50] C.S. Sharp, O. Shakernia, and S.S. Sastry. A vision system for landing an unmanned aerial vehicle. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1720–1727 vol.2, 2001.

[51] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing, 3rd Edition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.

[52] S. Saripalli, J.F. Montgomery, and G. Sukhatme. Vision-based autonomous landing of an unmanned aerial vehicle. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 3, pages 2799–2804, 2002.

[53] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti, and S. Longhi. A single-camera feature-based vision system for helicopter autonomous landing. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1–6, 2009.

[54] D.G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999.

[55] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[56] Pedro J. Garcia-Pardo, Gaurav S. Sukhatme, and James F. Montgomery. Towards vision-based safe landing for an autonomous helicopter. *Robotics and Autonomous Systems*, 38(1):19–29, 2002.

[57] S. Bosch, S. Lacroix, and F. Caballero. Autonomous detection of safe landing areas for an uav from monocular images. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5522–5527, 2006.

[58] Andrew Edie Johnson, James F. Montgomery, and Larry Matthies. Vision guided landing of an autonomous helicopter in hazardous terrain. In *ICRA*, pages 3966–3971. IEEE, 2005.

[59] Peter J Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

[60] John Stowers, Michael Hayes, and Andrew Bainbridge-Smith. Quadrotor helicopter flight control using hough transform and depth map from a microsoft kinect sensor. In *MVA*, pages 352–356, 2011.

[61] D. Kırcalı, F.B. Tek, and İ.K. İyidir. Ground plane detection using kinect sensor. In *21st Signal Processing and Communications Applications Conference (SIU)*, Girne, K.K.T.C., 2013.

Curriculum Vitae

Doğan Kırcalı was born on 20 April 1987, in İstanbul. He received his B.S. degree in Electronics Engineering and minor degree in Computing Systems in 2009 from Işık University with honors. He worked as a research assistant at the Department of Computer Engineering of Işık University between 2009 and 2010. He is working as a research assistant at the Department of Electric-Electronic Engineering of Işık University since 2010. The courses which he assisted include Introduction to Programming, Programming Workshop, Computer Organization, Microprocessors, Circuit Theory, Electrical Circuits, Electronics, System Dynamics and Control. His research interests are microcontrollers, robotics and computer vision. His personal interests are RC helicopters & planes and photography.

Publications

[1] İ.K. İyidir, F.B. Tek, and D. Kırcalı. Adaptive visual obstacle detection using monocular camera and ultrasonic sensor. *In 12th European Conference on Computer Vision* (ECCV2012),- *3rd Computer Vision in Vehicle Technologies Workshop* (CVVT2012), Florence, Italy, 2012.

[2] D. Kırcalı, F.B.Tek, and İ.K.İyidir. Ground plane detection using kinect sensor. In *21st Signal Processing and Communications Applications Conference* (SIU), Girne, K.K.T.C., 2013.