ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

ON-BOARD VISUAL ODOMETRY FOR CONTROL OF MICRO AERIAL VEHICLES

M. Sc. THESIS

Bulut KIRCA

Department of Mechatronics Engineering

Mechatronics Engineering Programme

AUGUST 2015

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

ON-BOARD VISUAL ODOMETRY FOR CONTROL OF MICRO AERIAL VEHICLES

M. Sc. THESIS

Bulut KIRCA
(518121004)

Department of Mechatronics Engineering

Mechatronics Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Erdinç ALTUĞ

AUGUST 2015

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

MİKRO HAVA ARAÇLARININ GÖRSEL ETKİN POZ TAKİBİYLE KONTROLÜ

YÜKSEK LİSANS TEZİ

Bulut KIRCA
(518121004)

Mekatronik Mühendisliği Anabilim Dalı

Mekatronik Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Erdinç ALTUĞ

AĞUSTOS 2015

Bulut KIRCA, a **M.Sc.** student of **ITU Institute of Science and Technology** student ID 518121004, successfully defended the **thesis** entitled "**ON-BOARD VISUAL ODOMETRY FOR CONTROL OF MICRO AERIAL VEHICLES**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Erdinç ALTUĞ** İstanbul Technical University	....
Jury Members :	**Asst. Prof. Dr. Gülay ÖKE** İstanbul Technical University	....
	Asst. Prof. Dr. Ayhan KURAL İstanbul Technical University	....

Date of Submission : **4 Aug 2015**
Date of Defense : **5 Aug 2015**

To my family,

FOREWORD

First, I would like to thank my advisor Assoc. Prof. Dr. Erdinç Altuğ for his continuous support and guidance. His friendly and at the same time professional way of supervision increased the satisfaction I get from this study.

Further, I would like to thank to İTÜ BAP for their financial support to this project.

Finally, I would like to thank my family for their unconditional love and the support they provided me. This study is dedicated to them.

August 2015

Bulut KIRCA
(Electrical Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

UAV **:** Unmanned Aerial Vehicle
DLP **:** Digital Light Processing
GPU **:** Graphics Processing Unit
CUDA **:** Compute Unified Device Architecture
CPU **:** Central Processing Unit
SLAM **:** Simultaneous Localization and Mapping
IMU **:** Inertial Measurement Unit
LIDAR **:** Lase Imaging Detection and Ranging
GPS **:** Global Positioning System
MAV **:** Micro Aerial Vehicles
VO **:** Visual Odometry
PID **:** Proportional Integration Derivation
EKF **:** Extended Kalman Filter

LIST OF TABLES

LIST OF FIGURES

Page

ON-BOARD VISUAL ODOMETRY FOR CONTROL OF MICRO AERIAL VEHICLES

SUMMARY

The usage of unmanned aerial vehicles (UAVs) is increasing in military, scientific, and civilian areas. With the increase in the number of unmanned aerial vehicles, autonomous navigation of UAVs is also gaining more volume. Among the various UAV types, the quadrotor platform has been used in many applications because of its hovering ability and mechanical simplicity. It is predicted that quadrotor will be one of the most common robotic platforms in the near future. Autonomous control of UAVs requires fast and accurate measurements of vehicle's pose and velocity in real-time. Moreover, for indoor applications, collusion free control of robots demands knowledge of environments 3D model. In this study, a visual odometry process proposed for unmanned vehicle applications. In contrast to conventional vison-based methods, the proposed method does not use visual features such as corners, edges or surfaces. In direct odometry approach, the motion estimation is obtained without using a feature descriptor or matching algorithm. Thus, the load on the processor decreased and the odometry system has better performance in real-time. For the real world application, camera's color and depth images are used. In order to minimize the error, an energy function has utilized.

Indoor autonomous vision-based UAV applications usually utilize passive stereo systems as a primary position sensor. However, the passive stereo systems are not completely capable of working in low textured areas, which are common in typical indoor environments. Here, we are interested in producing dense depth data in those areas where passive systems suffer from low textured nature of the environment. In this thesis, we use an active stereo system to obtain dense depth data in low textured areas. The active stereo system is implemented by using a DLP projector and stereo camera system. The pattern projected by structured light source is consist of 200 lines, which have random colors, positions and orientations to maximize ambiguity of pattern. At the end, the ambiguity of used pattern feeds our correspondence algorithm. The experimental results have also shown that the developed sensor system is supplying dense data regardless of the scene texture.

For the motion estimation process, we use direct visual odometry process. Direct visual odometry is presently one of the popular methods for motion estimation because of its ability to work on intensity values directly. The method uses all pixel data rather than visual features like corners, edges and surfaces which increases the efficiency of visual odometry software. In this study, we use depth and RGB image data from our custom RGB-D sensor. The motion between consecutive frames is calculated by minimizing intensity error function. In order to minimize the error, our method aligns consecutive frames in every iteration, in the end of alignment process, the source and reference frames have the closest viewpoint. The implemented odometry system is developed in C++ and tested with Freiburg RDB-D benchmark dataset. The experiments have shown that the developed odometry software is capable of calculating position changes

of camera in 6-DOF. The implemented sensor has been tested on commercially available quadrotor system for autonomous flight in typical indoor environment.

The implemented visual odometry system is used on Dji Phantom V1.2 for autonomous indoor flights. The controller of the system is a GPU powered computer named Jetson TK1. To achieve GPU-accelerated performance, the image processing modules of our system implemented in CUDA. Therefore, our system has better real-time performance in contrast to CPU based application. For control of quadrotor, we send PWM signals to flight control unit of Phantom from Jetson TK1.

In this thesis, we have presented a lower cost system for extracting depth map and estimating camera motion for autonomous control of indoor aerial vehicles. The proposed system supply dense depth data even in typical low textured indoor environments. Moreover, the odometry process computing position changes without using feature extraction and matching algorithms, since it computes the motion by working directly on all pixel intensities. This direct approach also decreases the processing load of whole odometry process. Besides the direct approach of odometry process, GPU-accelerated image processing modules also dramatically boost the real-time performance of our visual odometry unit. Nevertheless, since we utilize a dlp projector, our sensor system requires working in dark areas to prevent interlacing between projected pattern and environment color. Therefore, as a future work, our system could be composed with infrared cameras and projectors for operation in daylights. Moreover, the real-time performance of the system could be increased by using bucketing method, thus system does not operate on irrelevant points, which occurs out of the cameras' range.

MİKRO HAVA ARAÇLARININ GÖRSEL ETKİN POZ TAKİBİYLE KONTROLÜ

ÖZET

Günümüzde insansız hava araçlarının (İHA) birçok çeşidi bulunmaktadır. Özellikle takip ve inceleme uygulamaları için havada asılı kalabilme özellikleri nedeniyle rotor-tabanlı sistemlerin diğerlerine oranla büyük avantajları mevcuttur. Rotor-tabanlı sistemlerin birçok türü mevcuttur. Bu sistemler araçtaki mevcut rotor sayısına ve rotorların ne şekilde konumlandırıldığına göre farklılık göstermektedir. En çok bilinen klasik helikopter türü, bir ana rotor ve bir arka rotordan oluşan iki rotorlu hava taşıtıdır. Otogyro, Gyrodyne, tandem, yan-yana, eksenel, üç rotorlu, dört rotorlu ve heksakopter, oktokopter gibi çok rotorlu hava taşıtları da büyük ilgi görmektedir. Bu çalışmada özellikle manevra yetenekleri ve yük taşıma kapasiteleri göz önüne alınarak quadrotor hava aracı konfigürasyonu tercih edilmiştir. İnsansız hava araçları arama/kurtarma operasyonları, inşaat, onarım ve güvenlik gibi birçok alanda kullanılmaya başlanmışlardır. Dünyada birçok araştırmaya konu olan dört rotorlu insansız hava araçlarının (quadrotor) çevik yapıları nedeniyle günlük yaşamda en fazla kullanım alanı bulacak robotik sistemlerden birisi olacağı öngörülmektedir.

Otonom quadrotorların veya diğer otonom araçların bilinmeyen ortamlardaki keşif, navigasyon ve haritalandırma faaliyetlerine Eş Zamanlı Lokalizasyon ve Haritalandırma (SLAM) adı verilir. SLAM esas olarak sensörlerden elde edilen verilere bağlı olan, çok katmanlı bir işlemdir. Günümüzde SLAM uygulamalarında lazer uzaklık ölçüm, ataletsel ölçüm, GPS, 3D LIDAR, sonar ve bir veya iki kameradan oluşan sensörler kullanılmaktadır. Ancak bu sensörlerin ya maliyetleri yüksek olmakta ya da sağladıkları veri dış koşullara bağlı olarak önemli hatalar içermektedir. Bu nedenle birçok otonom quadrotor çalışmasında farklı sensörler denenmiştir. Bu çalışmada iç ortamlarda güvenilir ve hızlı çalışabilecek sensörün geliştirilip bu sensörle bir quadrotorun kontrolünün yapılması amaçlanmıştır.

Bu çalışmanın amacı iç ortamlarda çalışan mobil robotlarda kullanılabilecek görü tabanlı algılayıcıyı geliştirmektir. Ancak başta koridorlar olmak üzere iç ortamlar köşe noktaları, kenarlar ve düzlemler bakımından çoğunlukla zengin değildir. Görsel özellikler bakımından fakir olması nedeniyle iç ortamların güvenilir 3 boyutlu modeli oluşturulamamakta ve kamera hareketleri takip edilememektedir. Bu yüzden bu tür ortamlarda pasif stereo sistemler kullanılarak gerçekleştirilen robotik uygulamalar da güvenilir olmamaktadır. Diğer taraftan günümüzde popülerlik kazanan bazı 3 boyutlu sensörler, ortamdaki özellikleri kullanmak yerine kendi oluşturdukları görsel şekilleri kullanmaktadır. Bu sensör sistemlerinde bir veya daha fazla sayıda kamera ve yapısal ışık kaynağı bulunmaktadır. Yapısal ışık kaynağı ise kızıl ötesi, LED veya LCD projektörler olabilmektedir. Bu sistemlerde ışık kaynağı kamera görüş alanı içindeki cisimler üzerinde düzgün dağılmış özellikleri yansıtmaktadır. Bu sayede kamera tarafından alınan her karede ortamın yoğun bir derinlik haritası elde edilebilmektedir. Elde edilen derinlik haritası ise hem ortamın 3 boyutlu modelinin oluşturulmasında hem de çeşitli algoritmalar yardımıyla kamera hareketlerinin hesaplanmasında kullanılabilmektedir. Kinect günümüzde en yaygın olarak bilinen 3 boyutlu sensördür.

Bu çalışma kapsamında geliştirilen 3 boyutlu algılayıcı da, diğer aktif stereo uygulamalarında benzer şekilde, bir adet projektör ve 2 adet kameradan oluşmaktadır. Geliştirilen sensörde yapısal ışık kaynağı olarak ise mikro DLP projektör kullanılmıştır.

Aktif ve pasif stereo sistemler arasındaki en büyük fark, aktif stereo sistemlerinin kameraların yanında yapısal ışık kaynağı da kullanmasıdır. Birçok aktif stereo uygulamasında yapısal ışık kaynağı çevresine düzgün dizilmiş geometrik şekilleri yansıtmaktadır. Ancak düzgün dizilim kameralar tarafından sağlanan görüntüler arasındaki ilişkiyi basitleştirmekte elde edilen derinlik haritasının sık sık hatalar içermesine neden olmaktadır. Bu sorunun üstesinden gelmek amacıyla, bu çalışma kapsamında geliştirilen sensörde rastgele dizilmiş çizgilerden oluşan doku kullanılmıştır. Bu dokuda yer alan çizgilerin her biri rasgele renk, konum ve açılara sahiptirler. Oluşturulan şekildeki çizgilerin sayısının bulunmasında ise deneysel bir yaklaşım denenmiştir. Yapılan testlerin ardından 200 adet çizgi için stereo görüntü işleme algoritmalarının en iyi şekilde çalıştığı gözlenmiştir. Bunun yanında, geliştirilen sensör, iç ortamlarda sık karşılaşılan durumlar için test edilmiştir. Yapılan testte pasif stereo ve geliştirilen sensör düz duvarın önüne yerleştirilen bir cismin 3 boyutlu modelini oluşturmuştur. Test sonunda, pasif stereo tarafından oluşturulan derinlik haritasının sadece düz duvarın önündeki cismin kenar bilgilerini içerdiği görülmüştür. Geliştirilen sensör ise hem duvar hem de cismi içeren bir derinlik haritası oluşturmuştur.

Projenin temel hedeflerinden biri de görsel odometri yöntemini kullanarak kamera hareketlerini hesaplamaktır. Günümüzde robotik alanında popüler araştırma konularından biri olan görsel odometri için literatürde farklı yöntemler yer almaktadır. Birçok çalışmaya konu olan görsel odometri sürecinde, referans görüntü için, ortamdaki cisimler üzerinde köşe, kenar ve yüzey gibi çeşitli özellikleri seçilmekte ve bu özelliklerin bağıl yer değişimleri kamera tarafından sağlanan yeni görüntü yardımıyla hesaplamaktadır. Ancak her bir referans kamera görüntüsü için özelliklerin tespit edip ardından bu özellikleri yeni görüntüde bulması nedeniyle, geleneksel görsel odometri yaklaşımı işlemci üzerinde fazla yük oluşturmaktadır. Bu durum ile başa çıkmak amacıyla, son yıllarda, görsel odometri hesabının doğrudan ve görüntüdeki bütün pikselleri kullanarak yapılabileceğini gösteren çalışmalar ortaya çıkmıştır. Doğrudan görsel odometri denilen bu yöntemde piksel yoğunlukları kullanılmaktadır.

Geleneksel görsel odometri uygulamalarında hareket hesapları görüntüdeki kenar, köşe ve yüzey gibi görsel özelliklerin takibi ile yapılmaktadır. Bu yaklaşımda, özelliklerin tespit edilmesinde Haris, Shi-Tomasi, FAST, SIFT, SURF ve CENCURE gibi özellik tanımlama algoritmaları kullanılır. Bu algoritmalardan biri veya bir kaçı ile tanımlanan özellikler referans kamera görüntüsüyle karşılaştırılarak, eşleşmeler tespit edilir. Eşleşen özellikler takip edilerek kamera hareketleri hesaplanır. Doğrudan piksel yoğunlukları üzerinden çalışan odometri sistemleri ile karşılaştırıldığında, geleneksel uygulamalar özellik tespit etme ve eşleştirme şeklinde fazladan iki farklı adımı kapsamaktadırlar. Bu nedenle, performans açısından, doğrudan yöntemlere göre daha az verimli çalışmaktadırlar.

Kamera hareketlerinin doğrudan tüm pikseller kullanılarak hesaplanmasında ortamdaki özellikler yerine görüntüdeki bütün piksellerin verileri kullanılır. Çalışma kapsamında kullanılan kameralar renkli görüntü sağlamaktadır. Ancak odometri hesaplamalarında sadece piksel yoğunlukları kullanıldığı için renk bilgisine ihtiyaç duyulmamıştır. Bu amaçla kamera tarafından sağlanan renkli görüntülerden gri ölçekte yeni görüntüler elde edilmiştir. Geliştirilen hareket takip algoritmasında birbirini takip eden iki görüntü kullanılmaktadır. Bu görüntülerden birincisi referans, ikincisi ise kaynak görüntü olarak adlandırılmaktadır. Bu görüntülerin her biri hem renkli görüntü hem de ilgili derinlik haritasını içermektedir.

Ardışık görüntüler arasındaki hareket hesabı, bu görüntüleri hizalayarak yapılır. Hizalamanın tam doğru şekilde yapılması durumunda, referans ve kaynak görüntünün yoğunluk görüntülerinin birbiriyle bağlantılı pikselleri arasındaki yoğunluk farkı sıfıra eşit olacaktır. Ancak uygulamada ortamdaki hareketli cisimler veya sensörden kaynaklı hatalardan dolayı hizalama hesabını mükemmel bir şekilde yapmak mümkün olmamaktadır. Bu nedenle çalışmada en küçük kareler yaklaşımı ile bir hata fonksiyonu tanımlanmıştır. Bu hata fonksiyonu ardışık görüntülerde yer alan her piksel çifti arasındaki yoğunluk farkını kullanmaktadır. Ardından geliştirilen yazılım bu hata fonksiyonunu iteratif olarak belirli bir değerin altına hizalama yaparak düşürmeye çalışmaktadır.

Formüle edilen odometri algoritması, Visual Studio'da C++ programlama dili kullanılarak geliştirilmiştir. Geliştirilen yazılımda OpenCV Kütüphanesi görüntü işleme amacıyla kullanılmıştır. Bunun yanında matris işlemleri için Eigen, dosya sisteminin yönetimi ve çoklu görevler için ise Boost kütüphanesi kullanılmıştır. Üst seviye açık kaynak kodlu C++ kütüphaneleri olan Eigen ve Boost, odometri yazılımının performansını önemli ölçüde arttırmışlardır. Geliştirilen algoritmanın gerçek koşullarda test edilmesi için ise iki adet RGB-D veri seti kullanılmıştır. Bu veri setlerinden birincisi öteleme hareketlerini, ikincisi ise karmaşık hareketleri içermektedir. Kinect kullanılarak elde edilen bu görüntüler 640×480 çözünürlüğe sahiptir. Saniyede 30 kare çekilerek elde edilen görüntülerin elde edilmesi sırasında kameranın pozisyon değişimleri 100Hz ile çalışan hareket yakalama sistemi ile takip edilmiştir. Bu veri setinde yer alan iki hareket için de geliştirilen odometri yazılımı test edilmiştir. Yapılan testlerin sonuçları bölüm 3'te verilmiştir.

Bir quadrotorun daha önce bilinmeyen bir ortamda kontrolü eş zamanlı haritalandırma ve konumlandırmayı gerektirmektedir. Çalışma kapsamında geliştirilen 3 boyutlu algılayıcı, Dji firması tarafından üretilen Phantom V1.2 quadrotoru üzerinde test edilecektir. Yapılacak testlerde projektör, stereo kamera ve bilgisayar quadrotor üzerine yerleştirilecektir. Kullanılan bilgisayar kamera ve görüntü işleme algoritmalarını kullanarak quadrotorun konumunu takip edecektir. Bilgisayar tarafından yapılan konum hesaplamalarına göre, oto pilot sistemi araca ait motorları kontrol edecektir. Sensör ve bilgisayar kullanılarak modifiye edilen quadrotor bilinmeyen bir ortamda tanımlanacak başlangıç ve bitiş noktaları arasında hareket ettirilerek test edilecektir.

Bu çalışma kapsamında aktif stereo sistemi kullanılarak, köşe ve kenar gibi görsel özellikler bakımından fakir iç ortamların modellenebileceği gösterilmiştir. Bunun yanında kamera hareketlerinin doğrudan odometri yaklaşımı ile hesaplanabileceği ve bu yaklaşım ile odometri hesaplamalarının işlemci üzerinde oluşturacağı yükün de azaltılabileceği ortaya çıkmıştır. Çalışmada, literatürde yer alan doğrudan odometri uygulamalarının aksine ortamdaki cisimler değil aktif stereo sistemi tarafından yansıtılan dokular takip edilmiştir. Böylelikle hesaplamalarda ortaya çıkan hatalar azaltılmıştır. Geliştirilen sensörün ataletsel ölçüm birimleri ile tümleşik şekilde kullanılması ise hataları çok büyük oranda azaltabilir.

1. INTRODUCTION

Unmanned aerial vehicle (UAV) is the common name of the vehicles, which are used in various applications except passenger transport. They can be driven via a remote controller or work autonomously on a pre-defined job. The common usage areas of UAVs in military are target detection, mine detection, observing enemy activities and patrol [1]. Nowadays, aerial photography is the most common civilian application of UAVs. Furthermore, they work in agricultural areas, disaster fields and energy transmission lines. Both military and civilian application areas of UAV continuously increase day by day [2]. In recent years, autonomous UAVs also gained increased interest in robotics field.

Estimating the ego-motion of vision systems is a key issue for autonomous navigation of UAVs. It is also prerequisite for many other tasks like obstacle detection, augmented reality and simultaneous localization and mapping (SLAM). Motion estimation, referred to autonomous navigation of UAVs, could be performed with inertial measurement units (IMUs). However, data supplied by IMUs are distorted by drift and sensitive to environment conditions such as temperature or magnetic fields [3]. Wheel speed sensors are also used in motion estimation applications, yet they cannot provide reliable data in slippery terrains [4]. In addition to IMUs, laser imaging detection and ranging (LIDAR) systems are also suitable for indoor applications, and they provides accurate 3D range data in the point cloud form. Nevertheless, the movement of robots distorts the LIDAR data, as point clouds are belongs to different time intervals and accurate alignment of point clouds is a trivial task [5-6]. In outdoor applications, on the other hand, global positioning systems (GPS) are widely used in autonomous UAV applications. However, GPS measurements are sensitive to outside effects, and they cannot operate in indoor environments. Therefore, visual odometry process is relatively more advantageous for indoor applications, since it can provide proper motion data, which is less affected by drift. Estimating the motion from visual inputs can be achieved by feature-based or direct approaches. Feature-based approaches track visual features for calculating camera motion after extracting and detection steps. Direct approaches, on the other hand, do not require visual features. Hence, direct approaches are computationally economic, since feature extraction and detection steps of feature-based methods present relatively high load on computation unit.

Although vision based motion estimation systems has many advantages such as low cost and compact structure, they are not reliable in unstructured environments, which are common in indoor applications. Depth information of a scene also requires visual features, which are mandatory to feed correspondence algorithm [7]. However, typical indoor areas do not have enough visual features, such as corners, edges or surfaces, which are required for passive stereo systems. Unlike passive stereo systems, active stereo systems are capable of providing depth data of a scene, which is poor in terms of visual features. The key point is that active methods use their own light sources, besides the camera, to emit structured light into the scene. In this way, they produce dense depth data even if the scene has no visual features.

In this study, we combine active stereo and direct visual odometry approaches to estimate camera motions in unstructured environments. According to the above classifications, the technical details and experimental results of our approach are given in Section 2 and 3. In section 4, the application of our approach on a UAV is described. The thesis is concluded with future work description in Section 4.

1.1 Problem Statement

One of the most important requirements of robotics is finding the pose of the camera with respect to the world. In many applications using visual sensors, the relative position of reference (previous camera frame) and target (current camera frame) is required. Pose estimation is generally conducted using passive stereo systems and utilizing feature tracking algorithms. When the environment has sufficient visual features such as edges, corners and surfaces, challenge of pose estimation is taken by passive stereo process. The algorithms understand the motion of camera and compute the pose of camera as needed [7].

Motion estimation can also be achieved by using inertial measurement units (IMU) [3], wheel encoders [4] or laser distance measurement units (LIDAR). However, camera based motion estimation approaches are has numerous advantages. Especially, visual motion estimation’s accuracy is better than wheel encoder. Wheel encoders supply inaccurate data in slippery terrain, while visual odometry provide reliable motion calculations [4]. Moreover, IMU based estimations suffer from bias drift [3]. LIDAR sensors supply reliable input. However, using LIDARs in mobile systems cause deformation in point cloud, as the two consecutive point clouds belongs to different time intervals [6].

In conventional odometry applications, motion estimation is performed through tracking visual features like edges, corners and surfaces. For feature detection, traditional approaches uses Harris [8], Shi-Tomasi [9], FAST [10], SIFT [11], SURF [12] and CENCURE [13] algorithms. The process looks for the matches in consecutive frames and tracks the matching features for motion estimation. Therefore, the feature based odometry process has two additional steps called feature detection and matching, which makes this methods computationally more expensive than direct techniques. Direct approaches differently use all pixel data for motion estimation. For consecutive frames, motion estimation systems based on direct methods align frames and minimize the intensity difference between every pixel. Alignment ratio gives the relative motion between frames.

Usually indoor environments, such as corridors, do not have enough visual features to feed the feature tracking and correspondence algorithms. Therefore, the passive stereo systems are not reliable for motion estimation through direct approaches. In this case, given a low textured environment, the navigation of robots can be achieved through active stereo systems. Active stereo systems utilize a structured light source, which projects coded or random pattern. In this way, 3D modelling or estimating the change in camera pose can be obtained regardless of visual feature amount. Acquisition of best motion estimation can be obtained with Iterative Closest Point (ICP) algorithm, tracking features or direct visual odometry. In this study, we use direct visual odometry method to calculate camera motion.

1.2 Visual Pose Estimation

The navigation of Micro Aerial Vehicles (MAV) requires fast and accurate measurements of vehicle's pose and velocity. In this thesis, Visual Odometry (VO) process is utilized for ego motion estimation of a quadrotor. VO incrementally determines the position and orientation of the agent through the examination of the changes in image sequences from cameras induced by vehicle. In traditional VO applications, visual features, such as edges, corners or surfaces, are mandatory for calculation of change in camera pose. However, computational cost of feature extraction and matching is relatively higher. Our system uses direct approach to overcome processing cost of images. In direct methods, VO algorithm operates directly on pixel intensities rather than the features, which eliminates the computation cost and results in subpixel precision.

1.3 Fusing Depth and RGB Data for Odometry

The depth map of the camera environment is calculated with active stereo system featuring random pattern and a stereo camera rig. Active stereo system has been chosen for its capability to work in low textured environments and supply dense depth map. The RGB image from the left camera is used in visual odometry process. Nevertheless, the direct method is only interested in alignment of consecutive frames, thus we simply converts the RGB images to grayscale to feed our intensity error function. The value of intensity error function is based on the difference between grayscale values of consecutive image frames.

The depth data is used during the warping process of intensity image. At this step, the 2D intensity image is converted to a height map via projection function. The depth of the scene is calculated by image processing libraries of OpenCV. For boosting performance of the system, we use GPU accelerated libraries on Jetson TK1. The points exist in away from camera are omitted. The alignment of reference and source camera frame continues until the intensity error function is minimized. The details of alignment process is given in section 4.

1.4 Scope and Organization of the Thesis

In this study, the task of motion estimation via active stereo system and direct visual odometry process is undertaken. In section two, the active stereo system is described. This section starts with system overview, pattern generation and choice. The depth map acquisition and experimental results are given at the end of section. The experimental results contain comparative depth maps, which are obtained with passive and active stereo systems.

Section 3 is on computation of camera position using the direct visual odometry process. This section starts with overview and detailed formulation of direct visual odometry process. The section ends with overview of applications and experiments. We give details about implemented odometry software and datasets used in experiments. Finally, the results of experiments are given for translational and complex motion of camera.

2. DEPTH MAP ACQUISITION

In conventional odometry applications, the features of static world are tracked through the image frames for ego motion estimation. On the other hand, in order to use direct approach, dense depth map of scene is also required which is generated by finding correspondences in stereo image pair. However, the correspondence algorithms are fed by visual features of environment. Hence, unstructured indoor environments, such as corridors, are not suitable for passive stereo vision systems [7]. Therefore, we propose an active stereo system based on a stereo camera pair and a light source projecting random line patterns. Our custom RGB-D sensor utilizes a line pattern, which consists of lines with random orientation, position and colors. For ego motion estimation, we directly use projected active features to reduce the computational cost of on board visual odometry prediction and robust indoor navigation of agent.

Our custom RGB-D sensor consists of a micro dlp projector and a stereo camera system. In common applications of active stereo method, the light pattern involves uniform vertical or horizontal bars. This approach gives acceptable results for 3D scanner applications, yet correspondence problem is highly simplified for visual odometry applications because of the uniformity. Therefore, in this study, chosen projected light pattern consists of 200 lines with random orientation, position and colors. Hence, the ambiguity of pattern increases the complexity of correspondence problem.

2.1 RGB-D Cameras

RGB-D cameras novel sensor systems that provide color (RGB) images and depth (D) information for every related pixel. For capturing color images, these sensors use RGB cameras. The depth information, however, is calculated by different active vision methods.

2.1.1 Overview of RGB-D Cameras

RGB-D camera system can contain one or more digital cameras for taking RGB images, an infrared camera for sensing reflected infrared lights or a structured light source for emitting infrared or colored pattern. The digital camera provides color images encoded in RGB scale. The depth information is calculated by active or passive methods for every pixel in color image by using cameras with or without projector.

The widely known stereo camera systems are passive type of depth sensors. These systems utilize two color cameras, which have a certain transformation between each one. The depth calculations are achieved by finding corresponding points in image pair. The matching points have different position in each frame of image pair, thus the depth of the point can be calculated [7].

In contrast to passive stereo systems, active stereo systems not only detect the light reflected by environment, but also observe the light emitted by projector. The projector usually acts like a second camera of the system and its pattern directly feeds the correspondence algorithm. Hence, active systems are able to work in low-textured environments with low lighting condition, since they use their own light. Time of Flight (ToF) and projective stereo methods are most widely used depth estimation methods used in RGB-D cameras. In ToF based sensors, the light source sends pulses by switching the light source for very short time intervals at a certain frequency. The camera of the system observes light and computation unit resolves depth of scene based on the speed of light, and the distance of a point to the sensor [14]. On the other hand, projective active stereo systems have a similar working principle with passive stereo systems, but they use a light source rather than a second camera. In these systems, the light source emits a known pattern into the scene. The camera observes the projected pattern and computation unit finds the matching points between pattern and camera image. Using the position of matching points in both pattern and image, the depth of the scene can be calculated, since the transformation between projector and camera is known. In figure 2.1, schematic of an active stereo system is given. The given system has two mandatory part named IR light for projecting structured light and IR camera for capturing infrared images. The depth of the scene is calculated by mentioned active stereo process.

In recent years, RGB-D cameras, such as Microsoft Kinect and ASUS Xtion, become very popular in related robotics applications because of affordable price and performance. Both of the cameras utilize a system-on-a-chip developed by PrimeSense, which is based on projective active stereo technology. For emitting light into the scene, these cameras use an infrared light source and an IR camera that able to collect infrared light reflected by environment. Moreover, a RGB camera is exist in the system for collecting color data from the scene. In the end, the complete system supplies a color image and depth data for every pixel in RGB image. In figure 2.1, schematic of Kinect system architecture is given. Since these cameras have affordable price, and low-weight structure compared to ToF and stereo cameras, numerous applications are implemented by using them as primary RGB-D sensor [16].

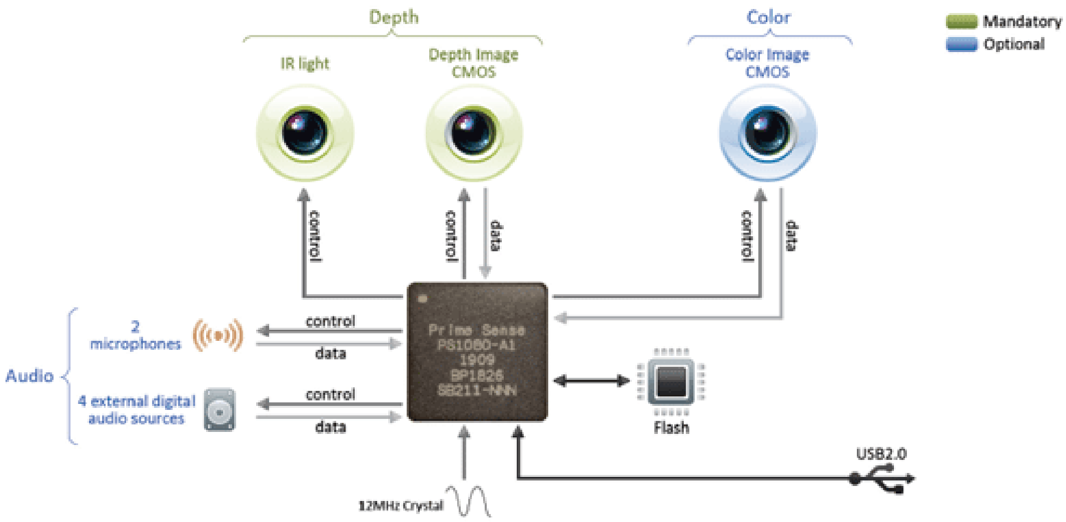

Figure 2.1 : Schematic of Kinect system architecture [15].

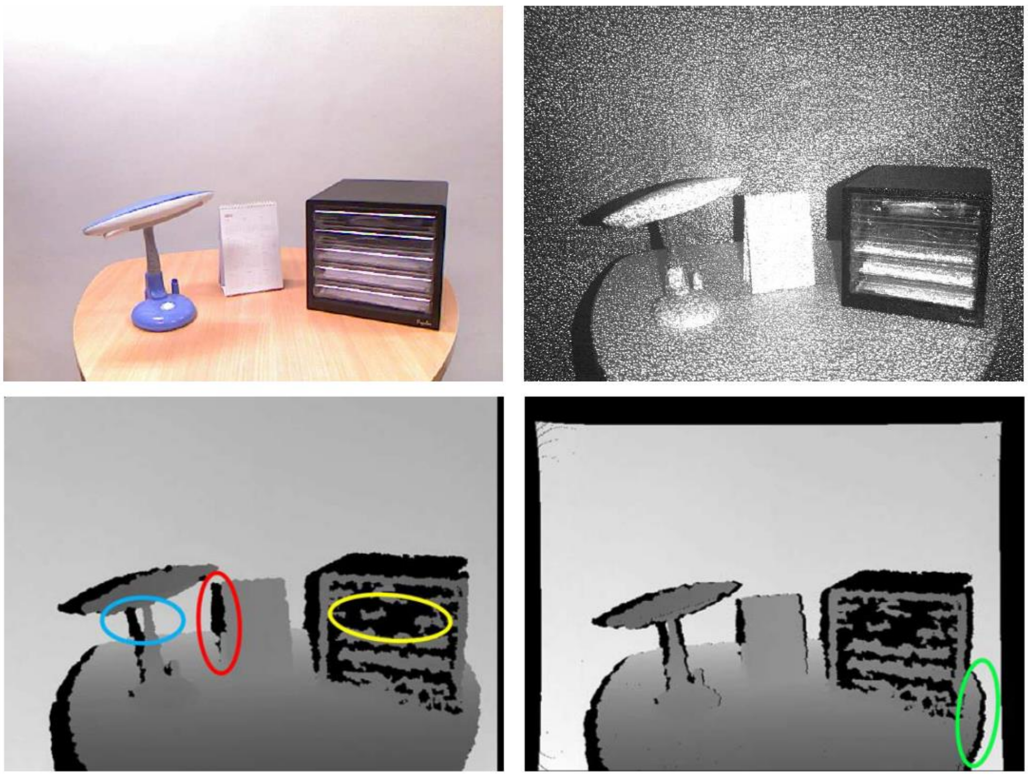

Figure 2.2 : Color, IR and depth images taken by Kinect Sensor [17].

In figure 2.2, the RGB and infrared images taken by Microsoft Kinect are given with related depth images. In depth image, the whiteness of the pixels indicates the depth level of the points. White pixels are further away, and gray pixels show the closer points. Moreover, Kinect cannot calculate the value of some points, which are black pixels in given images. The black pixels consist in the areas with no computable depth values, which is possible if the projected light is out of IR camera range.

2.1.2 Microsoft Kinect specifications

In this study, we have evaluated our algorithms with an RGB-D dataset and benchmark, which is generated by using Microsoft Kinect. The Microsoft Kinect uses projective stereo technology, as explained in [18], which is developed by PrimeSense. It provides RGB and depth images with 640×480 pixels resolution at a rate of 30 frames per second. A higher frame rate 60 frames per second can be obtained by reducing the resolution to 320×240 pixels [16].

The depth values are encoded as 16 bit unsigned integer values representing the depth in millimeters. According to the specification, the depth values range from 0.8m to 3.5m.

Figure 2.3 : Microsoft Kinect Sensor [16].

2.2 Pinhole Camera Model

The pinhole camera model is referred to mathematical relationship between 3D physical world and two-dimensional image. The model is used for mapping 3D points into a 2D plane [19]. This mapping, also called projection π, is stated by

$$\pi: \mathbb{R}^3 \rightarrow \mathbb{R}^2 \tag{2.1}$$

In this study, the depth acquisition modules utilize the pinhole camera model, and the model is given in figure 2.4. The model consists of a camera aperture defined as an infinitely small hole, also called pinhole, and an image plane. The mathematical relationship of the model describes perspective projection. The pinhole camera model does not include geometric distortions or blurring effects. Therefore, by first order approximation, only light rays falling through the hole and intersecting with the image plane are projected. The position of the pinhole is the optical center C of the camera. Moreover, the focal length f is referred to the distance between optical center and image plane.

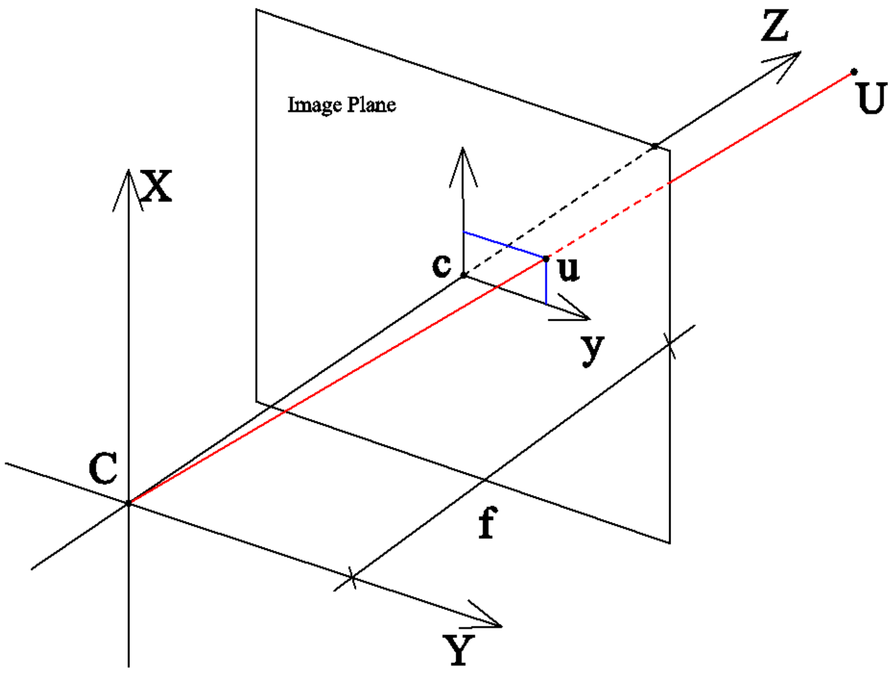

Figure 2.4 : Pinhole camera model.

In the physical world applications, the image plane lies below the optical center and not above of it. However, in the pinhole camera model this can be considered otherwise. The points feature in the image has a line between optical center and image plane. Coordinates of a point X is characterized by (X, Y, Z) in physical world, and the relevant 2D image point x is expressed by (x, y) in the below equations.

$$\pi(X, Y, Z) \rightarrow (x, y) \tag{2.2}$$

$$x = f_x \frac{X}{Z} + o_x \tag{2.3}$$

$$y = f_y \frac{Y}{Z} + o_y \tag{2.4}$$

The focal length of camera is mandatory for transformation between image data and physical world position. The distance from the camera (lens) center to the image plane (image sensor) is called the focal distance. Every pixel of the image plane does not have to be symbolized by quadratic equation, thus; two different focal lengths f_x and f_y are used. The focal lengths are calculated through scaling factors s_x and s_y in the respective direction, thus $f_x = s_x \cdot f$ and $f_y = s_y \cdot f$. The offsets o_x and o_y account for the fact that optical center C does not cross with the origin of the image coordinates.

By utilizing focal lengths and origin of the image, for a depth value Z, the real world coordinates of an image point x can be calculated by using following inverse equations:

$$\pi^{-1}(x, y, Z) \longmapsto (X, Y, Z) \tag{2.5}$$

$$X = \frac{x - o_x}{f_x} Z \tag{2.6}$$

$$Y = \frac{y - o_y}{f_y} Z \tag{2.7}$$

$$Z = Z \tag{2.8}$$

The intrinsic parameters (f_x, f_y, o_x, o_y) can be identified by a basic camera calibration strategy [7]. All along the calibration, derived parameters like sensor skew or radial distortion coefficients are calculated. The sensor skew explains for a sensor not mounted parallel to the camera lens. The radial distortion coefficients show the distortion of the image because of the lens. Before the camera images are processed by the presented stereo algorithms, these effects are neutralized. Thus, the pinhole camera model holds.

2.3 Pattern Selection

In this thesis, the developed RGB-D sensor system utilizes a LED projector for projecting randomly distributed line patterns. In stereo camera applications, depth map of the environment is generated by finding corresponding points in image pair. On the other hand, indoor environments usually consist of low-textured parts, which require using structured lights. For structured light projection, we choose LG PH300 mini projector for its compact size, lightweight and embedded battery. This projector has HD resolution of 1280×720 pixels and work 2.5 hours with its built-in battery.

Figure 2.5 : Projected light pattern.

Conventional structured light approaches use vertical bars with different frequencies. However, such applications suffer from simplified correspondence problem. This simplification causes large uncertain areas in the resulting depth map of the environment. Here, we are interested in optimizing the performance of correspondence algorithm in low textured areas and producing dense depth map. Thus, for this project, employed line patterns must be random in the manner of color, position and orientation in order to avoid causal coincidences, that is, ambiguity in the stereo process. In figure 2.5, produced light pattern is given.

2.4 Disparity Space

Consider a rectified image pair with baseline B and an upper triangular camera intrinsic matrix consist of the focal length f and the principle point $c = (c_u, c_v)^T$. Without any loss of generality, the left image assumed to be the origin of coordinate system. A point $p = (x, y, d, 1)^T$ is an element of disparity space, where $x = u - c_u,\ y = v - c_v$ and $d = u - u_r$ is disparity; the difference between the u-coordinate in the left image and its relevant coordinate in the right image. For this rectified image pair, the depth of an image point can be measured with:

$$Z = Bf/d. \tag{2.9}$$

2.5 Implementation of RGB-D Sensor and Experimental Results

The 3D acquisition software is developed in C++ using Visual Studio. For stereo image processing, OpenCV libraries are utilized. Since calculating stereo images affect overall performance, we use GPU accelerated image processing libraries. The stereo images are captured with USB 3.0 stereo camera. We utilize block-matching algorithm with tunable SAD windows size and disparity range parameters. The stereo images are captured with LI-USB30-V024STEREO camera of Leopard Imaging. Stereo camera consists of two Aptina MT9V024 sensors. The image of camera is given in Figure 2.6.

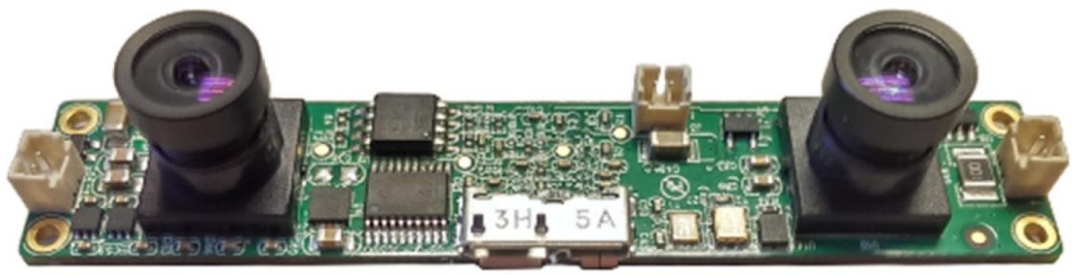

Figure 2.6 : The stereo camera.

The resulting depth data of the scene should be dense for accurate alignment of the consecutive frames. To compare performance of developed RGB-D system, we use passive and active stereo system for the same scene. Experimental results belongs to developed RGB-D sensor is given in figure 2.7. As shown in experimental results, the depth map of unstructured scene has large uncertain areas (black) in case of passive stereo system. These uncertain areas are exist in the location of behind wall, which could be considered as common part of the typical indoor environment. Therefore, passive type of application is not reliable for direct visual odometry process. However, the developed active stereo system is able to calculate depth of unstructured part of the scene by using the projected textures. This dense data make it possible to calculate motion of the camera in typical indoor environments. In this experiment, the developed active stereo system operates in the dark, since the projector of the system interlaces with the visible light. For operation in daylight, infrared camera and projector is possible replacement for our system.

Figure 2.7 : The reference frame on the left, corresponding disparity map obtained with passive stereo (second) and active stereo (third).

3. VISUAL ODOMETRY

3.1 Introduction

The navigation of Micro Aerial Vehicles (MAV) requires fast and accurate measurements of vehicle's pose and velocity. Moreover, for indoor navigation applications, collusion free control of a MAV demands knowledge of environment's 3D model. In this thesis, Visual Odometry (VO) process is utilized for ego motion estimation of a quadrotor. VO incrementally determines the position and orientation of the agent through the examination of the changes in image sequences induced by vehicle. In traditional VO applications, visual features, such as edges, corners or surfaces, are mandatory for calculation of camera pose. However, computational cost of feature extraction and matching is relatively higher. Our system uses direct approach to overcome processing cost of images. In direct methods, VO algorithm operates directly on pixel intensities rather than the features, which eliminates the computation cost and results in subpixel precision.

3.2 Feature-Based Methods

Although this study focuses on direct VO, it is worth mentioning feature-based approaches. In order to estimate motion, feature-based methods uses visual features such as edges, corners and surfaces. The motion estimation through this approach has four fundamental steps: feature detection, feature matching, feature tracking and motion estimation. In addition, principle pipeline of a sample feature based motion estimation process is given in figure 3.1. This kind of odometry processes can use one or more of the feature detection algorithms like Harris [8], Shi-Tomasi [9], FAST [10], SIFT [11], SURF [12], CENCURE [13]. The features exist in both reference and source images are employed in tracking step, which exploits one of the optic flow estimation algorithms like famous Lucas-Kanade Method [21]. After tracking all points, the method removes the points with different flow in a local neighborhood. Finally, the system estimates the camera motion and extracts the key points. It is well known that the computation cost of feature-based methods essentially comes from the feature detection, matching and tracking steps. Therefore, in this study, we focus on direct technique for obtaining a system, which is able to work efficiently on standard CPU's.

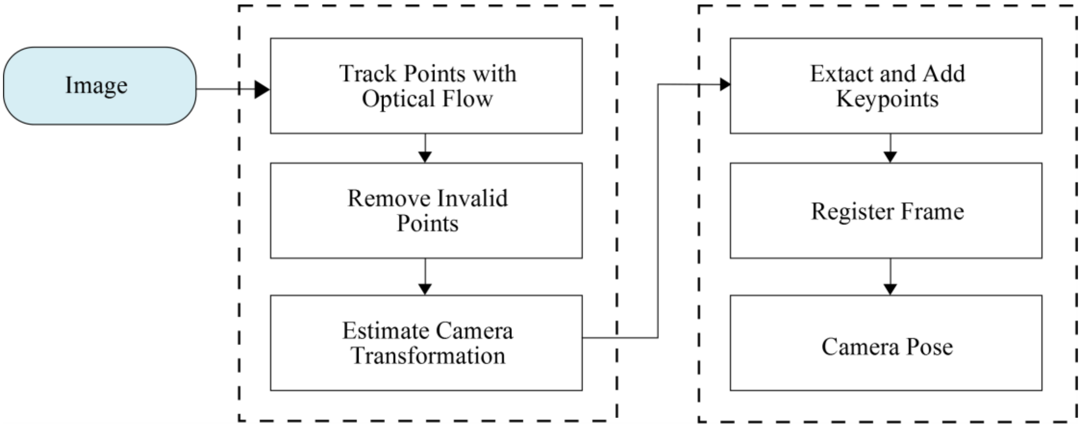

Figure 3.1 : Pipeline of a sample visual odometry process.

3.3 Rigid Body Motion

In this section, a brief introduction to the properties and modeling of the motion of rigid bodies in physical world is given. The distance and orientation of any two points exist in a rigid body is never change in the physical world during the motion of the body.

More consistently, a rigid body motion is a map g:

$$\boldsymbol{g}: \mathbb{R}^{\mathbf{3}} \to \mathbb{R}^{\mathbf{3}}; \boldsymbol{x} \mapsto \boldsymbol{g}(\boldsymbol{x}) \tag{3.1}$$

maintaining the distance and the orientation enclosed by two points p and q:

$$\begin{aligned} \|p - q\| = \|g(p) - g(q)\| \quad & \forall\, p, q \in \mathbb{R}^3 \\ g(p) \times g(q) = g(p \times q) \quad & \forall\, p, q \in \mathbb{R}^3 \end{aligned} \tag{3.2}$$

The rigid motion map g is also called a special Euclidean conversion. The combination of each of rigid motion map conversions in three-dimensional Euclidean space extends the special Euclidean group SE(3).

Since the properties of distance and orientation between any two points do not change, we can use it to show the motion of a rigid body in a compact way. The orientation and position change of one point having a connected to the coordinate system is acceptable to determine the motion of the all points of the object.

This conversion is consistently with depending on some reference coordinate system and can be used to determine a rotational and a translational part. The rotation variations, the orientation of the object coordinate frame and the translation moves it in space. In all, a rigid body motion has 6 degrees of freedom, both translation and rotation has three degrees of freedom.

There are numerous ways to show rotational part of a rigid body motion. One of the most frequent statement is a 3×3 orthogonal matrix R_t the rotation matrix. Every rotation matrix resides in the special orientation group SO(3). Besides a 3×3 orthogonal matrix R_t the rotation matrix, quaternions and a composition of a rotation angle and axis are also widely used. The translation is shown by vector $t \in \mathbb{R}^3$. The elements of t states the translation together with the x, y and z-axis. We can define a rigid body motion g as a composition of a rotation matrix from SO(3) and a translation vector from $\mathbb{R}^3$. This can be formulized by means of a 4×4 matrix G

$$G = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{3.3}$$

The inverse of the rigid body motion is:

$$G^{-1} = \begin{bmatrix} R^T & -R^T t \\ 0 & 1 \end{bmatrix} \tag{3.4}$$

The motion of a point p with homogenous coordinates $(x, y, z, 1)^T$ using g can be shown as multiplication of two matrices:

$$g(p) = g(G, p) = G \cdot p = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ r_{21} & r_{22} & r_{23} & r_{24} \\ r_{31} & r_{32} & r_{33} & r_{34} \\ r_{41} & r_{42} & r_{43} & r_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.5}$$

i.e. with $p' = g(p) = (x', y', z', 1)^T$

$$x' = r_{11}x + r_{12}y + r_{13}z + t_x \tag{3.6}$$

$$y' = r_{21}x + r_{22}y + r_{23}z + t_y, \tag{3.7}$$

$$z' = r_{31}x + r_{32}y + r_{33}z + t_z. \tag{3.8}$$

Numerous rigid body transformation can be combined using the matrix definition and by left multiplying ensuing motions. The identity rigid body motion, stating no motion, is represented by $R = I$ and $t = 0$, i.e. $g_{identity}(p) = p$.

The motion definition as vector t is canonical, since the three elements of the vector correspond to degrees of freedom. On the contrary, the definition of a rotation as matrix is not canonical. This definition consists of nine parameters yet just three degrees of freedom. The other parameters are constrained because of the orthogonality condition of the matrix and the limitation to norm equal to one for every row and column vector. This appoints six constraints, and just three free parameters are left.

A rigid body transformation can be shown minimally a definition which can be achieved by utilizing the parameters of its related Lie algebra $\mathfrak{se}(3)$. This kind of minimalist definition is advantageous during resolving the parameters by a numerical optimization method. All transformation matrices residing in the Lie group $SE(3)$ represents a rigid body transformation includes a definition in its related Lie algebra with a 6×1 parameter vector $\theta = (v^T, \omega^T)^T$ where $v = (v_1, v_2, v_3)^T$and $\omega = (\omega_1, \omega_2, \omega_3)^T$ are the rotational and translation velocities correspondingly.

The rigid body transformation g can be computed by utilizing its own Lie algebra parameters θ adopting the exponential map:

$$\begin{gathered} exp: \mathfrak{se}(3) \to SE(3); \theta \longmapsto g, \\ G(\theta) = e^{\hat{\theta}} \end{gathered} \tag{3.9}$$

Here θ is accepted as twist and is the 4×4 matrix given below:

$$\theta = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 & v_1 \\ \omega_3 & 0 & -\omega_1 & v_2 \\ -\omega_2 & \omega_1 & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.10}$$

The operator $[x]_x$ builds 3×3 skew symmetric matrix in distinction to a 3×1 vector $x = (x, y, z)^T$, i.e.

$$[x]_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}. \tag{3.11}$$

The matrix exponential $e^{\hat{\theta}}$ holds a closed form formulization

$$e^{\hat{\theta}} = \begin{bmatrix} e^{[\omega]_x} & V_v \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{3.12}$$

where $e^{[\omega]_x}$ is calculated by utilizing Rodrigues' description:

$$e^{[\omega]_x} = I + \frac{\sin(\|\omega\|)}{\|\omega\|}[\omega]_\times + \frac{1 - cos(\|\omega\|)}{\|\omega\|^2}[\omega]_\times^2 \tag{3.13}$$

and V is

$$e^{[\omega]_x} = I + \frac{\sin(\|\omega\|)}{\|\omega\|}[\omega]_\times + \frac{1 - cos(\|\omega\|)}{\|\omega\|^2}[\omega]_\times^2 \tag{3.14}$$

The inverse to the exponential map can be named as the logarithm map

$$\begin{gathered} log\colon SE(3) \to \mathfrak{se}(3); g \mapsto \theta \\ \theta = \log(G) \end{gathered} \tag{3.15}$$

The identity motion is calculated for $\theta = 0$.

3.4 Least Squares

The approach of least squares is a widely used method to calculate the most possible parameters of a model by utilizing noisy measurements. In the next section, we give information about the general idea and mathematical statements behind the least squares method. Then, derivation of the mathematical statements from a probabilistic perspective and the postponement to the weighted least squares technique.

3.4.1 General

Least squares method has a definition $f(x, \theta)$ with a determined group of parameters $\theta(\theta_0, \theta_1, \dots, \theta_{m-1})^T$, for every $x(x_0, x_1, \dots, x_{n-1})^T$ to measurements $\hat{f}(\hat{f}_0, \hat{f}_1, \dots, \hat{f}_{n-1})^T$ by minimizing the squared error existing among the measurements and the model. However, the measurements consistently do not fit the least square model because of the noise or simplifications of model. Therefore, the error is never be zero. We can show this by,

$$E_{LS}(\theta) = \left(\hat{f} - f(x,\theta)\right)^T \left(\hat{f} - f(x,\theta)\right)$$
$$= \sum_{i}^{n} \left(\hat{f} - f(x_i,\theta)\right)^2 = \sum_{i}^{n} (r_i(\theta))^2 \qquad (3.16)$$

In equation 3.16, $r_i(\theta) = \hat{f}_i - f(x_i,\theta)$ is the i^{th} residual. The best parameters θ_{LS} are acquired by resolving below equation.

$$\theta_{LS} = \underset{\theta}{argmin}\, E_{LS}(\theta). \qquad (3.17)$$

The optimum value is determined by computing the partial derivatives of E_{LS} and solving them by equalizing to zero.

$$\frac{\partial E_{LS}(\theta)}{\partial \theta} = \sum_{i}^{n} \frac{\partial r_i(\theta)}{\partial \theta} 2 r_i(\theta) = 0 \qquad (3.18)$$

To solve this formulization, we should own as many measurements as unknowns, at worst, in θ to acquire a result. In real world applications, there are much more measurements than unknowns resulting in an over constrained mathematical statements.

In the certain situation, where $f(x_i,\theta)$ is linear in the parameters θ, i.e. $f_{lin}(x_i,\theta) = A(x_i)^T\theta$, dealing with 3.17 is named as linear least squares. $A(x_i)$ is an random function of x_i of dimension m. The result for linear least squares can be achieved in closed form. Substituting $f_{lin}(x_i,\theta)$ into equation 3.18 gives:

$$\sum_{i}^{n} A(x_i)^T\left(\hat{f} - A(x_i)\theta\right) = 0. \qquad (3.19)$$

This statement can be reorganized to

$$\sum_{i}^{n} A(x_i)^T A(x_i)\theta = \sum_{i}^{n} A(x_i)^T \hat{f}_i \quad \textbf{(3.20)}$$

and edited in matrix representation:

$$A(x)^T A(x)\theta = A(x)\hat{f}. \quad \textbf{(3.21)}$$

The parameter vector θ is the optimum result to these normal statements.

For definitions $f(x_i, \theta)$ with non-linear dependence on θ the method is named as non-linear least squares. $r_i(\theta)$ should be linearized utilizing a first order Taylor series method at a point α for $\theta = \alpha$, to achieve a linear dependence for the parameters θ.

3.5 Overview of Method

In conventional VO applications, motion estimation is accomplished by extraction of features from images [22, 23] while direct methods use all pixel data. Direct approach estimates the motion by using intensity values of all pixels without resorting the features as an intermediate representation of the image. The camera motion is tracked using alignment of two consecutive image frames so that pixel intensity dissimilarities are minimized. Direct methods have numerous distinctive properties, which make them effective in real world applications. Sub-pixel accuracy, locking and dense recovery of shape properties are prominent features of direct VO approach [20].

Direct VO is the one of the popular methods for motion estimation because of its ability to work on intensity values directly. In this study, we use depth and RGB image data from our custom RGB-D sensor. These images are utilized for estimating the agent's motion. The majority of process is developed in Visual Studio using C++ with the support of OpenCV libraries. Eigen Library, a C++ template library, is employed for performance required linear algebra, matrix and vector operations [34]. For multithread and file system related tasks, Boost library is utilized [35]. The pipeline of implemented VO process is given in figure 3.2.

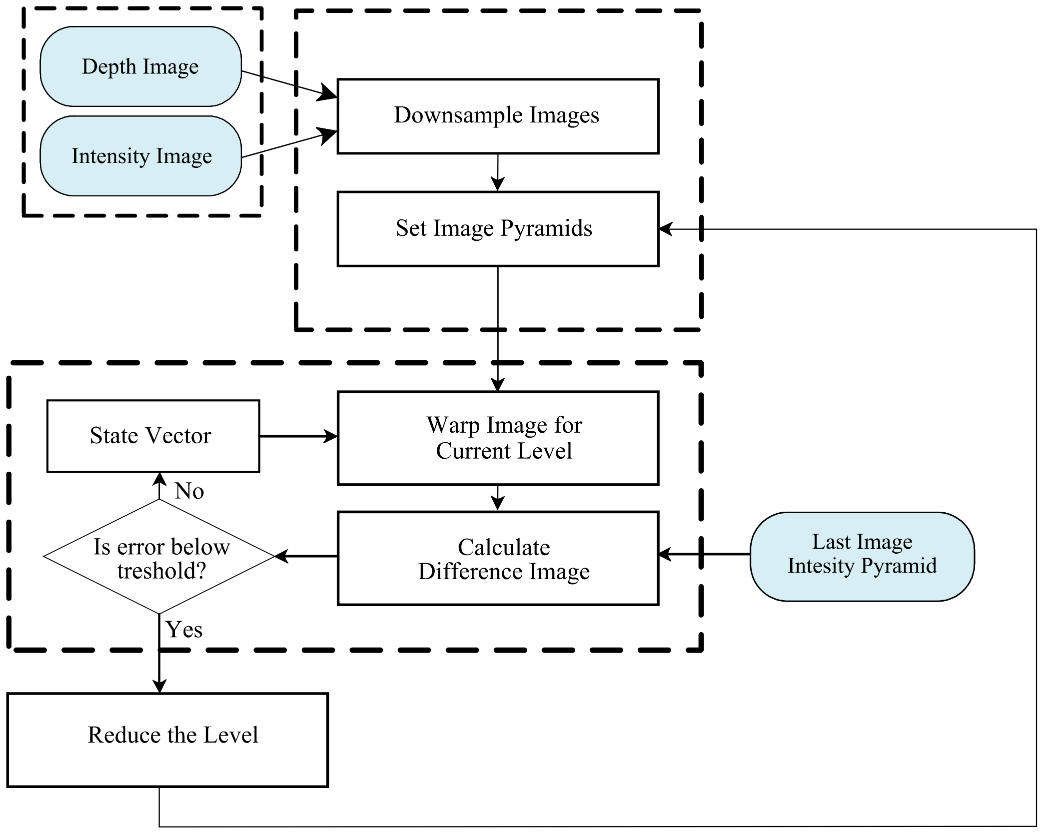

Figure 3.2 : Visual odometry pipeline.

3.5.1 Direct visual odometry

Direct VO process, in contrast to feature-based methods, uses whole image data. To find the motion between consecutive frames, it adjusts images through minimization of an error metric. Given the exact motion of the camera, two consecutive images can be transformed into identical viewpoint. In this case, difference between intensity values is equal to zero for all pixels. However, in reality, the intensity difference is never zero because of the dynamic objects, sensor noise or other effects. Even so, the error is still assumed minimum for the correct estimation.

Now, consider two consecutive frames of camera observations C_{t-1} and C_t, where C_{t-1} is reference frame and C_t is source frame. Each camera observation has a RGB and associated depth image, which are represented by I_{RGB} and h. Since we are only interested in intensity difference, the grayscale values of each pixel in I_{RGB} can simply be used. The grayscale images are defined as:

$$I = \frac{I_R + I_G + I_B}{3} \tag{3.22}$$

Figure 3.3 : Reference and source camera frame (top), related intensity image (bottom).

Let reference grayscale frame be given with I_0 where intensity of a point $\boldsymbol{p}$ is $\boldsymbol{I}(\widetilde{\boldsymbol{p}}) \in \mathbb{R}$, and the point is defined as:

$$\widetilde{\boldsymbol{p}} = \begin{bmatrix} x + c_u \\ y + c_v \end{bmatrix} = \begin{bmatrix} u \\ v \end{bmatrix} \tag{3.23}$$

With a slight abuse-of-notation, we hereafter use $\boldsymbol{I}(\widetilde{\boldsymbol{p}}) := \boldsymbol{I}(\boldsymbol{p})$. After rigid-body motion $G(\theta)$ between t_0 and t_1, intensity of the point becomes $\boldsymbol{I}'(\boldsymbol{p}')$, and the change in camera pose raises a small increment of pose parameters $\Delta\theta$ for an initial θ. The unknown motion between consecutive frames can be calculated by increasing the photoconsistency, which can be formalized as:

$$\Delta\theta^* = \underset{\Delta\theta}{argmin} \sum_{p \in \Omega} \left\| I'\big(w(p;\ \theta + \ \Delta\theta)\big) - I(p) \right\|^2 \tag{3.24}$$

In equation 3.24, Ω is a subset of pixel coordinates of interest in the reference frame. Moreover, $w(\cdot)$ is a warping function, which depends on the motion parameters we investigate. The current estimation of motion parameters are updated after every iteration by means of an additive rule:

$$\theta \leftarrow \theta + \Delta\theta \tag{3.25}$$

The iteration process repeats until the error converges or the termination criteria, e.g. maximum number of iterations, has been met. Equation 3.25 is the standard formulation of Forward Additive (Lucas-Kanade) algorithm [21]. There is also an efficient variation of Lucas-Kanade image alignment algorithm which is called Inverse Compositional (IC) algorithm. The IC algorithm proposes two modifications on computation of error in order to significantly improve the performance. The first adjustment is that the IC algorithm has a compositional update rule rather than additive one, thus it compounds the incremental estimates of $\Delta\theta$. Secondly, the IC algorithm switch the roles of reference image I and source image I'. Under the IC algorithm, the modified form of equation is used:

$$\Delta\theta^* = \underset{\Delta\theta}{argmin} \sum_{p \in \mathrm{I}} \left\| I\big(w(p;\ \Delta\theta)\big) - I'(w(p;\theta)) \right\|^2 \tag{3.26}$$

In equation 3.26, regardless of the form of warping function and parameters, an optimization problem raises. Using Taylor expansion, we obtain:

$$\Delta\theta = (J^T J)^{-1} J^T e, \tag{3.27}$$

where J is defined as:

$$J = (g(p_1)^T, \dots, g(p_m)^T) \in \mathbb{R}^{m \times p} \tag{3.28}$$

In equation 3.28, m is the number of the pixels and p is the number of parameters which is equal to $|\theta|$. Each g is $\in \mathbb{R}^{1\times p}$and is computed as:

$$g(p) = \nabla I(p) \frac{\partial w}{\partial \theta} \tag{3.29}$$

∇I is the image gradient along the u- and v- directions, and can be defined as:

$$\nabla I = (I_u, I_v) \in \mathbb{R}^{1\times 2} \tag{3.30}$$

Finally, the error image is:

$$e(p) = I'\big(w(p; \Delta\theta)\big) - I(w(p; \theta)) \tag{3.31}$$

At the next iteration of the optimization algorithm, parameters of the motion model are updated via the IC rule given by:

$$w(p, \theta) \leftarrow w(p, \theta) \circ w\big((p, \Delta\theta)\big)^{-1} \tag{3.32}$$

3.6 Algorithm

For a reference frame I with a corresponding depth image and a source frame later camera motion I', we look for to approximate the parameters of camera translation to find a value for which equation 3.3 has a minimum point. The warping statement is shown by:

$$w: (\mathbb{P}^3 \times \mathbb{R}^6) \to \mathbb{R}^2 \tag{3.33}$$

$$w(p, \theta) = \pi(\Gamma \mathrm{T}(\theta) \Gamma^{-1} p) + c \tag{3.34}$$

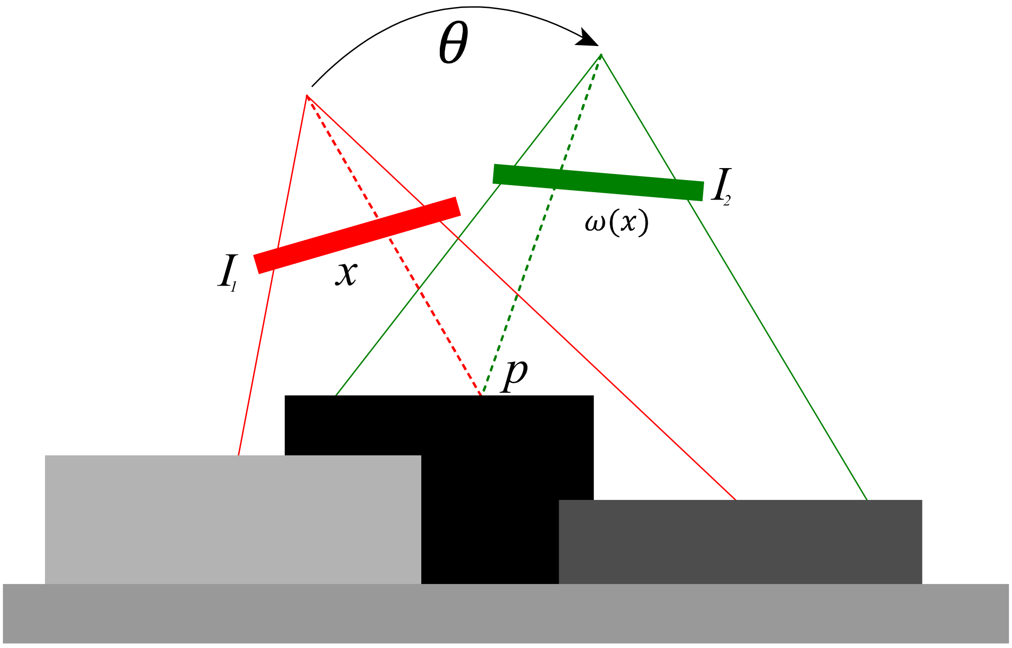

Figure 3.4 : Calculation of pixel coordinates in I2 for a camera motion θ.

In equation3.34, $\pi(\cdot)$ is the projection matrix handles two goals. The first goal of projection matrix is achieving homogenous division to recalculate the point for Euclidean space. The second goal is choosing the first two components of the normalized vector associated with the $x -$ and $y -$ coordinates. After all, to acquire 2D pixel position in the image coordinate system we put the principle point c back.

3.7 Jacobian of the Warp

It is well know that for applying a direct approach it is required to have an analytic representation of the Jacobian complies with the parameters for the identity $\theta = 0$. Utilizing the Lie algebra formulization of rigid body motions, i.e. twist, $\theta = \left(\omega_x,\ \omega_y, \omega_z, v_x, v_y, v_z\right)^T \in \mathbb{R}^6$, we acquire the Jacobian of the warping statement in eq. 10 for every point $\boldsymbol{p}$ as:

$$
\nabla I \frac{\partial w}{\partial \theta}\bigg|_{\theta=0} = g(\boldsymbol{p}) = \begin{pmatrix} -fI_y + \frac{\alpha y}{f} \\ fI_x - \frac{\alpha x}{f} \\ yI_z - xI_y \\ I_z\beta \\ I_y\beta \\ \frac{\alpha\beta}{f} \end{pmatrix}^T \in \mathbb{R}^{1\times 6}, \tag{3.35}
$$

In equation 3.35, $\nabla I = (I_x, I_y)$ is the image gradient, $x = c_u - u, y = c_v - v, d = u - u_r$, with

$$
\begin{aligned} \alpha &= xI_x + yI_y \\ \beta &= d/B \end{aligned} \tag{3.36}
$$

For m pixels, we substitute the parameters of eq. 3.36 into an $m \times 6$ matrix and achieve an revision of parameters $\Delta\theta$ by dealing with the normal equations in equation 3.27. The calculated value of the pose is changed after every iteration utilizing the inverse compositional update rule which is shown by:

$$
\begin{aligned} w(p;\theta) &\leftarrow w(p;\theta) \circ w(p;\Delta\theta)^{-1} \\ &:= \mathrm{T}(\theta)\exp\left(-\widehat{\Delta\theta}\right), \end{aligned} \tag{3.38}
$$

In above equation, $\hat{\cdot}: \mathbb{R}^6 \to \mathfrak{se}(3)$, and $exp: \mathfrak{se}(3) \to SE(3)$ indicates the matrix exponential in which $\exp(x)^{-1} = \exp(-x)$.

3.8 Error Minimization and Robust Ego Motion Estimation

The least square method is sensitive to outliers. Therefore, to achieve dependable enough estimation, it is required to replace the least square method with a robust error function. Here, the choice of error function is made experimentally.

The experiments have shown that the bi-weight function performs best among several cost functions. The bi-weight function for a residual $\tau \in \mathbb{R}$ is formulated by:

$$\rho(\tau_i;\tau) = \begin{cases} (1-(r_i/\tau)^2)^2 & if\ |r_i| \le \tau; \\ 0 & otherwise. \end{cases} \tag{3.39}$$

To achieve 95% asymptotic efficiency, we set the cutoff threshold τ to 4.6851. The threshold requires normalized residuals with unit deviations. For this purpose, we use a robust estimator of the standard deviation. For m observations and p parameters, the robust standard deviation is calculated by:

$$\hat{\sigma} = 1.4826\left[1 + \frac{5}{m-p}\right] median|r_i| \tag{3.40}$$

The constant 1.4826 is used to obtain the same efficiency of the least squares under Gaussian noise, while $\left[1 + \frac{5}{m-p}\right]$ is used to normalize for small data. In practice, $m \gg p$ and the small data constant disappears.

Finally, given the list of residuals is given by:

$$r_i = I'\big(w(p_i;\Delta\theta)\big) - I(p_i;\theta) \tag{3.41}$$

We compute a robust estimate of the standard deviation $\hat{\sigma}_r$ using eq. (3.40) and calculate the weight per residual as $w_i = \rho\left(\frac{r_i}{\hat{\sigma}_r}\right)$. By integrating the weights into an $m \times m$ diagonal matrix W, we may obtain an approximated value of the parameters at every iteration by calculating weighted normal equations:

$$\Delta\theta = (\mathrm{J^TWJ})\mathrm{J^TWe} \tag{3.42}$$

3.9 Appearance Variations

Utilizing a powerful pixel wise error approach does not guarantee the modelling of common appearance changes for a concrete data. Different algorithms in the image alignment research consider the appearance variations. A frequently applied way is to the gain + bias model, which grant affine appearance variations. Regularly, affine variations are considered as per image patch. It is, on the other hand, achievable to benefit a rise in efficiency by considering bias and affine gain globally for every pixel. Therefore, the reference image intensity can formulized as:

$$I(p_i) = b + (1 + g)I'(p_i). \tag{3.43}$$

In equation 3.43, b is the additive bias and g is the multiplicative gain. According to the inverse compositional algorithm, we obtain:

$$\Delta b + (1 + \Delta g)I(p_i) = b + (1 + g)I'(p_i). \tag{3.44}$$

By including appearance variations, the Jacobian of the warp turn into an 8-vector achieved by adding the derivatives w.r.t. gain and bias. Those results are the pixel intensity and a constant correspondingly. After all, appearance variation elements are revised with:

$$g \leftarrow (g - \Delta g)\eta, \text{ and } b \leftarrow \frac{b - \Delta b}{\eta}, \tag{3.45}$$

here

$$\eta = 1 + \Delta g. \tag{3.46}$$

Other techniques to illumination changes consists pf pre-filtering, or tracking the output of high dimensional statement of intensity, like descriptors.

3.10 Additional Implementation Details

The developed direct odometry method does not use complicated parameter setting or particular heuristics. The only modifiable parameters are the stereo algorithm parameters. The stereo algorithm parameters rely on the dataset. As explained in section 2, we utilize a block-matching stereo. The modifiable stereo parameters are SAD window size and disparity range. In order to deal with large motions and increase the performance, we downsample the intensity images and build image pyramids. Even so, the disparity images are not scaled to avoid interpolation across occlusion boundaries. Rather that scaling disparity images, the height map calculates at the full resolution. Every level of the pyramid is resized with a coefficient of $\frac{1}{2}$ of the previous image resolution. The intensity image frames are smoothed with a Gaussian or Box filter prior to scaling. All interpolation procedures in this study are achieved by a bilinear interpolation kernel. Convergence is calculated if the norm of the approximated parameters is less than 1×10^{-6}, variation in parameters is below 1×10^{-8} or a maximum number of iterations is reached. The maximum number of iterations is fixed to 300 iterations on the full resolution intensity image, and 50 iterations for all other level of the image pyramid.

Calculating the standard derivation of errors turns into a trivial operation due to the periodic requirement for computing their median at each iteration when dealing with a large number of data. To reduce this, the use of histograms can be utilized to calculate the median. Even though this approximation might improve efficiency, but the certainty of this approximation relies on the resolution of the histogram.

Rather than that, our approach calculates the median with a yielding algorithm and follows the variation in this computation along iterations. If the variation in calculated standard deviation is come below the threshold (1×10^{-6}), our developed software realize the calculation and do not compute it later. The approximation of median often resides within the first 10 iterations.

After all, we achieve a considerable efficiency development by ranging the baseline rather that the disparity for every pyramid level. Therefore, rather than bisecting disparities per pyramid level, we couple the baseline.

The efficiency developments given improve the efficiency of the approach obviously when utilizing a dense group of pixels. When utilizing a pixel selection scheme, nonetheless, the adjustments are imperceptible.

3.11 Experimental Details

3.11.1 Visual odometry software

In this study, the motion of camera is calculated via implemented VO software, which aligns consecutive frames of camera and minimizes the photometric error between intensity images. Since the operating system of onboard computer is Linux, whole odometry process is developed in C++ by using Visual Studio for debugging and building activities.

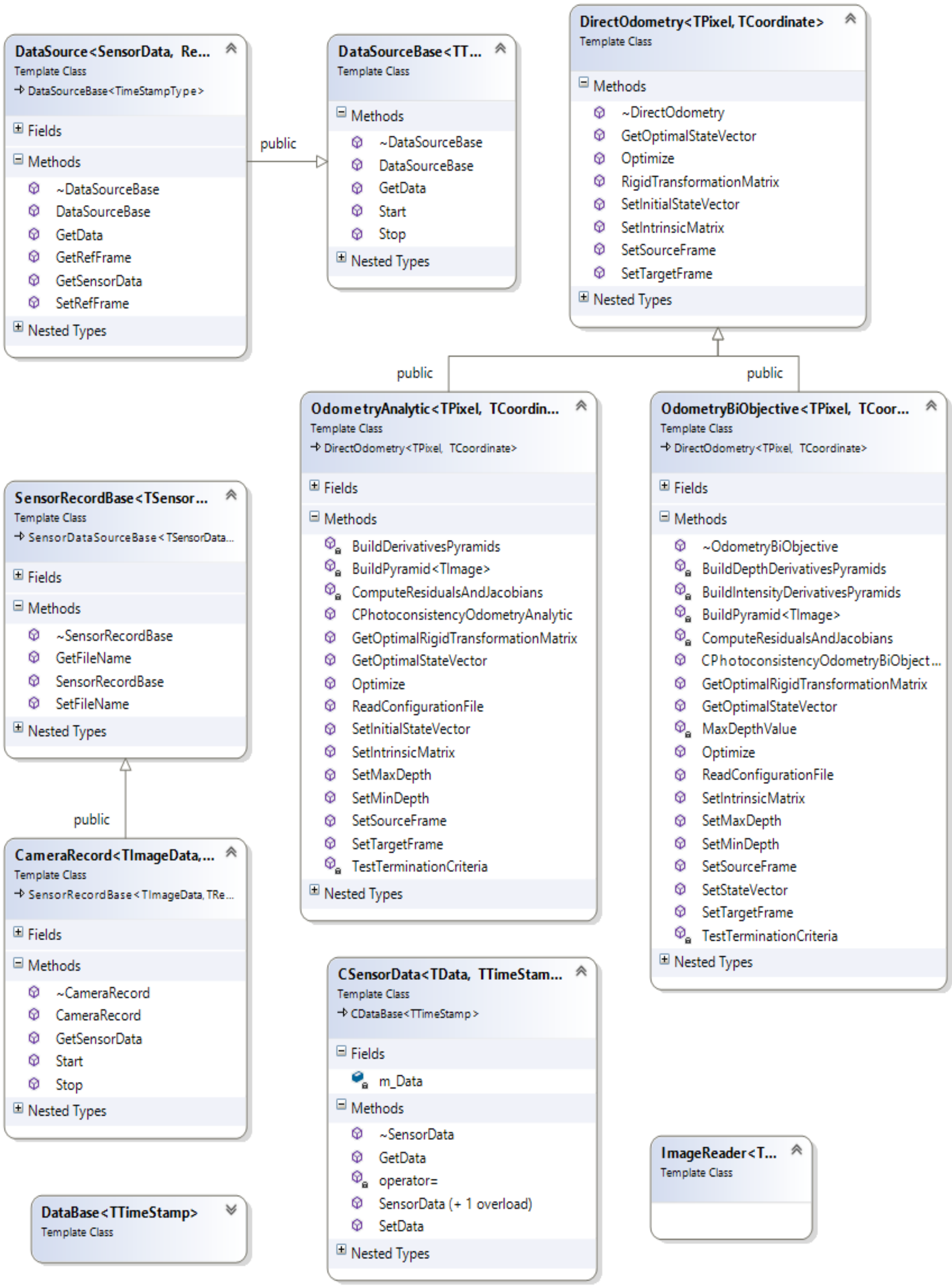

Figure 3.5 : Class diagram of the visual odometry software.

To get the efficient real-time performance, OpenCV 2.4.9 libraries are utilized for image processing tasks. Moreover, for the matrix calculations, we employ the Eigen 3.2.2 libraries. Eigen is an open source high-level library of template headers for linear algebra related operations. Finally, we used Boost 1.57 libraries for file system operations and multithreading. Using Eigen and Boost libraries has significantly increased the efficiency of software. In figure 3.5, the class diagram of VO is given and the pipeline of software is given in figure 3.2. As can be seen from class diagram and the pipeline, the software downsamples each intensity images to a certain level. After downsampling, the intensity images are warped and the photometric error is minimized iteratively.

3.11.2 RGB-D benchmark dataset

The performance of VO software evaluated via RGB-D benchmark dataset. The dataset consist of full resolution (640×480) RGB images and depth images at 30 fps. The image sequences are recorded with handheld camera with unconstrained 6-DOF motions in office environment. The dataset also contains ground-truth trajectory, which is generated by motion capture system [24]. Sample color and depth images from dataset are shown figure 3.3 and 3.4.

Figure 3.6 : Depth image from Freiburg dataset [24].

Figure 3.7 : RGB image from Freiburg dataset [24].

3.11.3 Stereo camera data

In this thesis, we combine our visual odometry system with stereo machine vision cameras produced by Leopard Imaging. This camera is UVC (universal video class) compliant. Nonetheless, the raw data obtained by cameras encoded in a GR Bayer format but dispatched as YUYV. The raw image captured by leopard camera is shown in figure 3.8, which necessitate decoding ahead computer vision algorithms can be used.

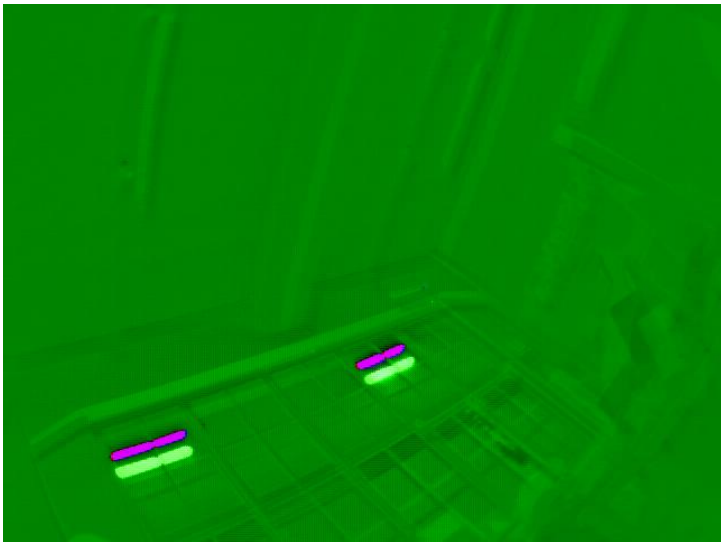

Figure 3.8 : Raw camera output of the stereo camera.

To get image frames from stereo camera, we have utilized libv4l2, video for Linux. After that, the raw data is interpreted into RGB rather than RG-Bayer pattern. A GR pattern works by storing all pixel values on one layer of the image in contrary to an RGB frame, which consist of a layer for red, blue, and green correspondingly. Thus, the GR Bayer pattern has a resolution much higher than the standard RGB image. After utilizing the method captured by leopard imaging for decoding GR-Bayer pattern, the image frame is accurately decoded. The derived stereo image is given in figure 3.9.

Figure 3.9 : Resulting image after decoding raw data from stereo cameras.

3.11.4 Experimental results

In order to test the performance of the visual odometry software, the ground truths of two image sequences are prepared. The first dataset consist of translational camera motion and the second dataset has complex camera motion. During the visual test, we have used all image frames, thus the VO software has calculate the motion at the 30 fps. The validation is achieved by intersecting the timestamp of each camera and ground-truth values.

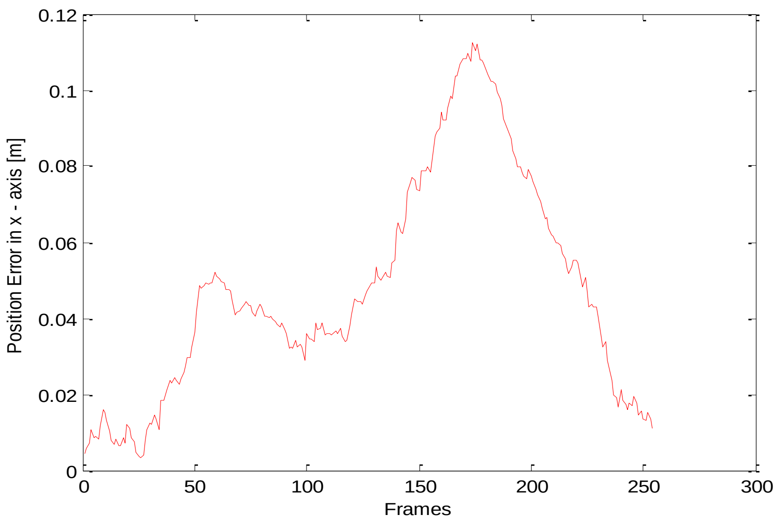

Figure 3.10 : Position error in x-axis.

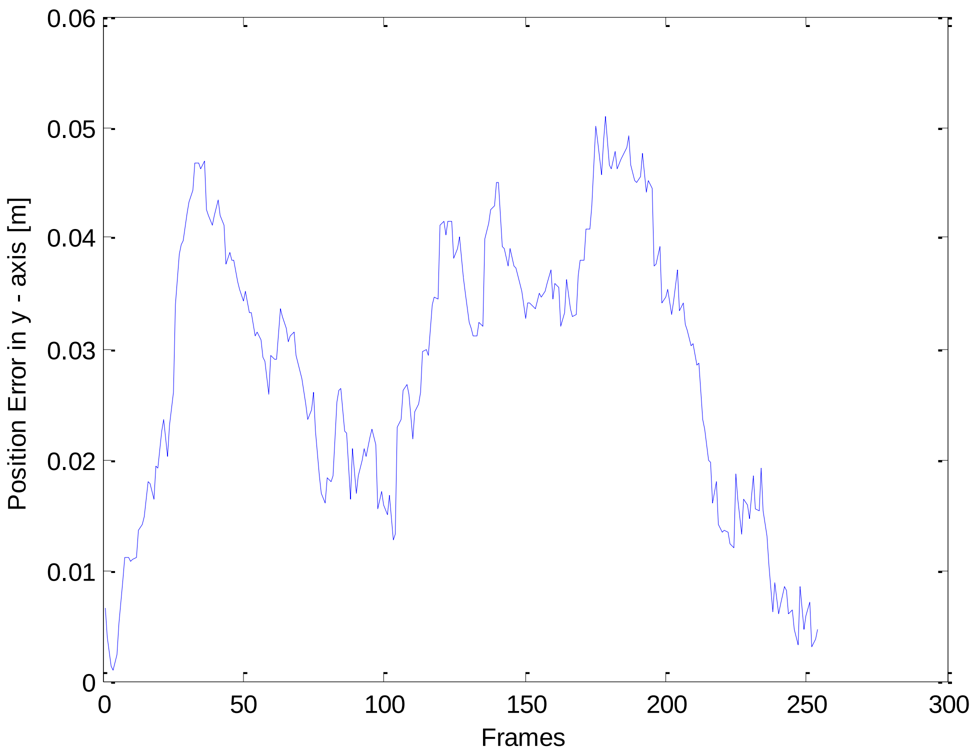

Figure 3.11 : Position error in y-axis.

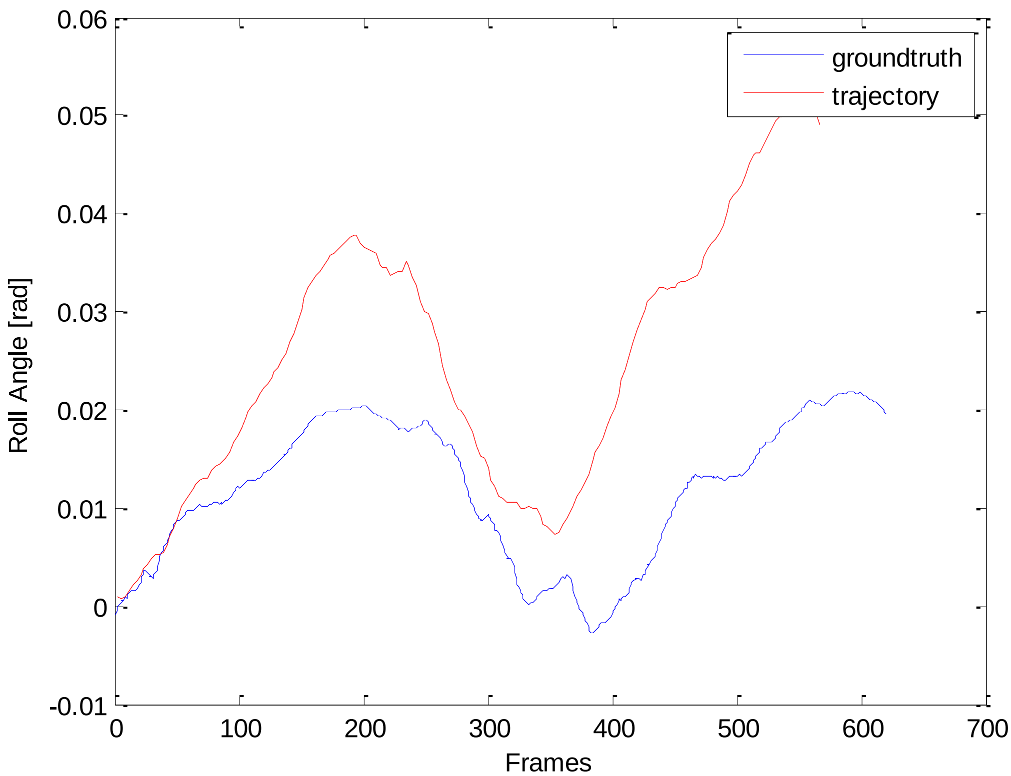

Figure 3.12 : Orientation error in roll-axis.

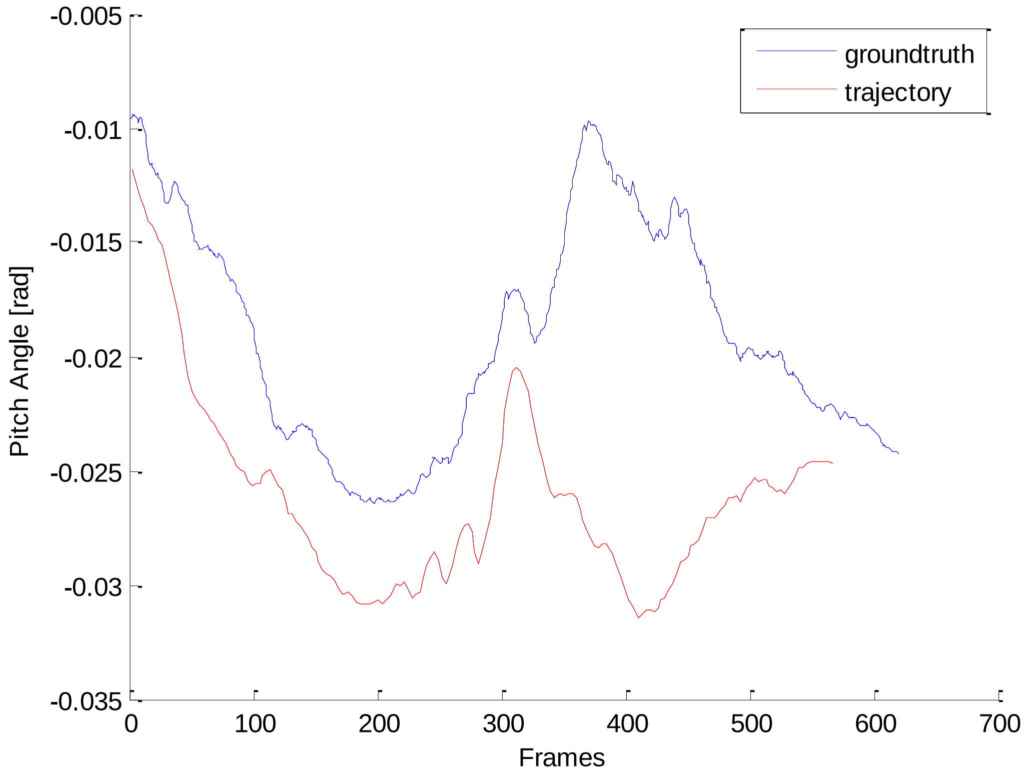

Figure 3.13 : Orientation error in pitch-axis.

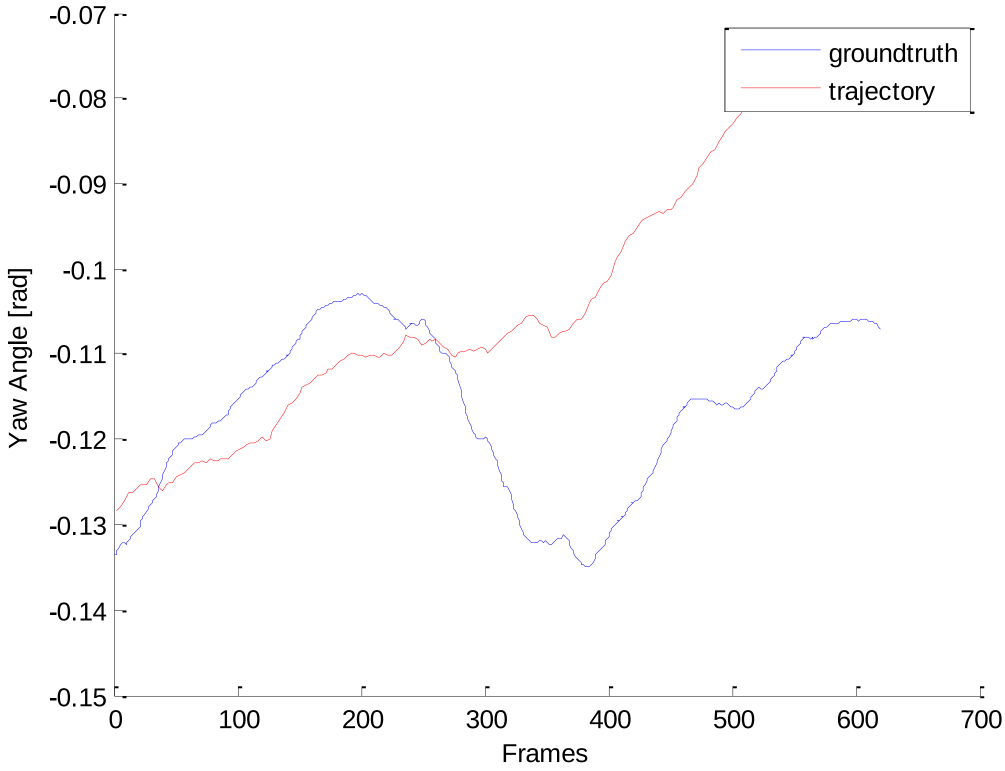

Figure 3.14 : Orientation error in yaw-axis.

4. QUADROTOR IMPLEMENTATION

4.1 Introduction

Quadcopters are rotary wing aircraft with four rotors [1]. The speed of the rotors entirely regulates the lift and attitude. This is on contrary to helicopters, which alter the composition of the blades of the main rotor over a mechanical structure. Hence, a quadrotor has fewer mobile components, which reduces its construction and servicing requirements. In contrast, the precise control of the rotation speed of the rotors, ea. stabilization is more difficult. As all rotor aircraft, quadcopter can take off vertically, hover and fly at low speeds. Figure 4.1 illustrates the system of the rotors and an acceptable choice to select the local coordinate frame, the coordinate system can be chosen randomly.

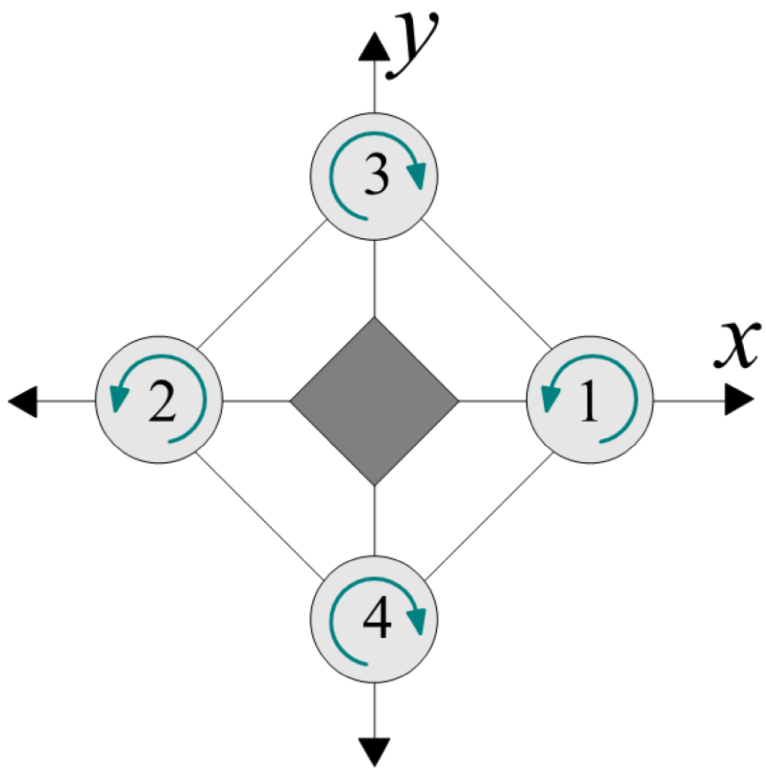

Figure 4.1 : Quadcopter rotor setup and example coordinate system.

In the given quadrotor setup, two opposite rotors form a pair rotating in the same direction. The other pair rotates in the opposing direction. To change height the speed of all four rotors is increased simultaneously or decreased. Rotation around the x-axis (roll) is achieved by reducing the speed of the left rotor and increasing the speed of the right one, or vice versa. Similarly, different rotation speeds of the front and rear rotor cause a rotation around the y-axis (pitch). Differences in the speed of the rotor pairs cause a rotation around the z axis (yaw).

4.2 DJI Phantom V1.2

The DJI Phantom V1.2 is a medium sized quadrotor with a width and length of 35cm and height of 19cm. Therefore, it is small enough to fly in indoor environments, and it has a payload capacity of 680 gr. Figure 4.2 shows a side-view of Phantom without visual odometry systems.

Figure 4.2 : Dji Phantom version 1.

4.3 Master Board – Jetson TK1 Development Kit

The visual odometry process and control of quadrotor are implemented for Jetson TK1 development kit. Jetson is a full-featured platform for Tegra K1 embedded applications and has 190 CUDA cores. It is a powerful solution for vision and robotics applications. Moreover, Jetson's power consumption is between 1 to 5 watts. In this study, Jetson utilized for depth map acquisition, direct odometry process and EKF based control of quadcopter. Therefore, it is called as master board in this study. The operation system of Jetson is Ubuntu and it has USB 3.0 port, which is required for stereo camera. It is also capable of generating PWM signal to control motors.

Figure 4.3 : Jetson TK1 development kit.

4.4 Quadrotor Implementation

The control of quadrotor in unknown environment is achieved by the data supplied by sensors and the localization is can be achieved in the existence of a map. The traditional approach to simultaneous localization and mapping is simultaneous localization and mapping (SLAM). The common application of SLAM is using a laser sensor to map the objects and the localizing the vehicle by means of created map. SLAM is the problem of performing both mapping and localization at the same time. In this study, the quadrotor determine the own coordinate frame according to a fixed coordinate frame and the navigation loop use this process as feedback.

SLAM problem is more complicated than localization, since the algorithm calculates agent’s state and the other points of the map. Therefore, the state space model of the system is larger than the other applications, which only perform localization.

Applied on board of the quadrotor the visual odometry approach requires an efficient implementation. Furthermore, it has to be integrated with the RGB-D camera drivers, the control and the data fusion software. The system is implemented on the Robot Operating System (ROS) middleware. The main reasons for this choice are that the

ROS has support for Jetson TK1 platform, package management, visualizers, and OpenCV interface packages.

Figure 4.4 depicts the overall structure of the software components and their communication paths. The camera driver gives the stereo image pairs and depth map generator provides RGB_D images to the tracker. The camera tracker implements the presented visual odometry approach.

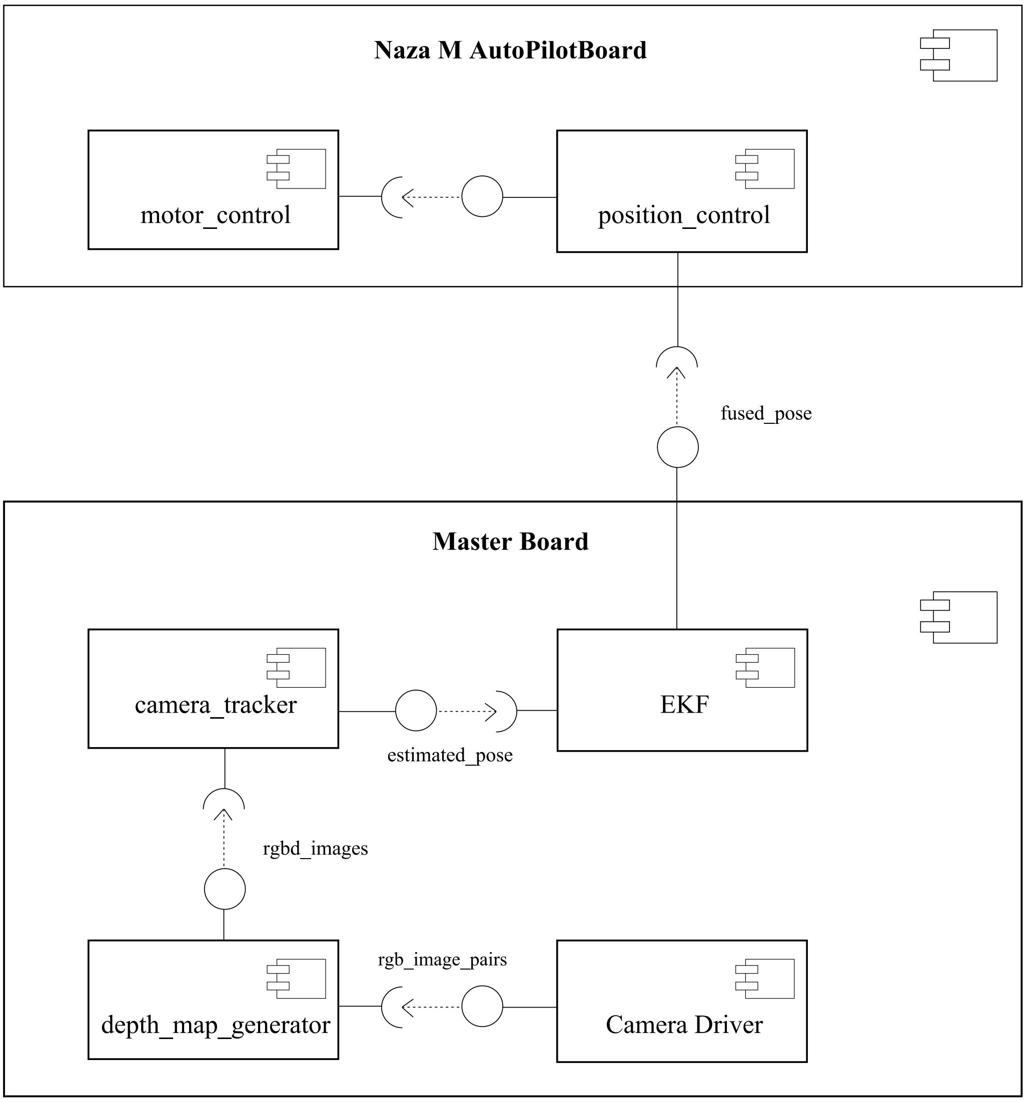

Figure 4.4 : The software components and their integration.

The distributed EKF framework uses the pose estimates to correct the state predictions. These predictions are calculated from the IMU data at 1 kHz on the Autopilot Board of the Phantom. Finally, the position controller uses the state estimate of the EKF, which includes the fused data from the visual odometry and the IMU, to stabilize the quadcopter. The one main contribution of this thesis is the camera tracker component. It implements the camera motion estimation from an RGB-D image pair using the theoretical foundations described in section 3. The pseudo code of the core function is given in section 3. It applies a coarse to fine scheme, to estimate the camera motion. Starting with a low-resolution version of the images allows compensating for larger motions and accelerates computation. At every level l multiple iterations are performed. In every iteration k the Jacobian matrix J, the weighting matrix W and the residuals r are computed.

4.5 Kalman Filter and Extended Kalman Filter

Kalman filter is estimator for quadratic problems and it is applied to linear systems, which are affected by white noise or has measurements containing white noise [38]. Kalman filter is one of the most significant practical inventions of twentieth century. The main idea is that Kalman Filter estimates the state of the system by observing many measurements belong a time interval rather than single measurement. Considering the estimated state resides a time interval, Kalman Filter's approximations are accurate. Extended Kalman Filter (EKF) is used for non-linear systems. In this study, we propose to use EKF for fusing IMU and odometry data.

4.6 EKF-Based State Estimation

The developed direct visual odometry system is accompanied by an IMU with accelerometers and rate gyroscopes. Combining IMU data with the developed direct odometry system improve the robustness of the visual-frame to frame-motion calculation. Furthermore, the combined system is able to provide pose estimations in case of a failure, e.g. encountering with close objects and dramatic changes in lighting. In this thesis, we incorporate IMU and odometry data through an indirect Extended Kalman Filter (EKF). Considering Kalman framework desires certain measurements, employing virtual pose calculations is not achievable because visual computations demonstrate respective motions between consecutive frames. Therefore, our system utilizes previous motion estimations, which grants integrating relative measurements. This method is assigned to as stochastic cloning.

Five parameters: position $\boldsymbol{p}$, rotation $\overline{\boldsymbol{q}}$ as unit quaternion, velocity $\boldsymbol{v}$, gyroscope biases $\boldsymbol{b}$ and accelerometer biases $\boldsymbol{c}$ form the state $\boldsymbol{x}$ of our system. Moreover, we add the position $\boldsymbol{p_d}$ and rotation $\overline{\boldsymbol{q}}_{\boldsymbol{d}}$ for the reference frame.

$$\boldsymbol{x} = [\boldsymbol{p} \quad \overline{\boldsymbol{q}} \quad \boldsymbol{v} \quad \boldsymbol{b} \quad \boldsymbol{c} \quad \boldsymbol{p_d} \quad \overline{\boldsymbol{q}}_d]^T \tag{4.1}$$

At the beginning, we fix $p_d = p$ and $\bar{q}_d = \bar{q}$ and the covariances from the current state variables for the related delayed states are accepted.

4.6.1 Propagation

In the below equations, the hat and tilde characters indicate computed and measured values relatively. The state is propagated by means of inertial measurements of $\widehat{\boldsymbol{a}}$ and $\widehat{\boldsymbol{w}}$, acceleration and rotation rate respectively. Computations are exploited by additive white Gaussian noise with variance $\boldsymbol{\eta_a}$and $\boldsymbol{\eta_\omega}$. The bias drifts are shown as arbitrary walk steps with noise densities $\boldsymbol{\eta_b}$ and $\boldsymbol{\eta_c}$. The correct states and related computed complements are derived as:

$$\begin{aligned}
&\dot{\boldsymbol{p}} = \boldsymbol{v} && \dot{\widehat{\boldsymbol{p}}} = \widehat{\boldsymbol{v}} \\
&\dot{\overline{\boldsymbol{q}}} = \frac{1}{2}\overline{\boldsymbol{q}}\overline{\omega} && \dot{\widehat{\overline{\boldsymbol{q}}}} = \frac{1}{2}\widehat{\overline{\boldsymbol{q}}}(\overline{\widetilde{\omega}} - \overline{\widehat{\boldsymbol{b}}}) \\
&\dot{\boldsymbol{v}} = \boldsymbol{R}_{\overline{q}}\boldsymbol{a} + \boldsymbol{g} && \dot{\widehat{\boldsymbol{v}}} = \boldsymbol{R}_{\widehat{\overline{q}}}(\widetilde{\boldsymbol{a}} - \widehat{\boldsymbol{c}}) + \boldsymbol{g} \\
&\dot{\boldsymbol{b}} = \eta_{\boldsymbol{b}} && \dot{\widehat{\boldsymbol{b}}} = 0 \\
&\dot{\boldsymbol{c}} = \eta_{\boldsymbol{c}} && \dot{\widehat{\boldsymbol{c}}} = 0 \\
&\dot{\boldsymbol{p_d}} = 0 && \dot{\widehat{\boldsymbol{p_d}}} = 0 \\
&\dot{\overline{\boldsymbol{q_d}}} = 0 && \dot{\widehat{\overline{\boldsymbol{q_d}}}} = 0
\end{aligned} \tag{4.2}$$

Here $\boldsymbol{R}_{\overline{q}}$ indicates the rotation matrix referred to $\overline{\boldsymbol{q}}$ and $\boldsymbol{g}$ is the gravity vector. The quaternions $\overline{\omega}$, $\overline{\widetilde{\omega}}$ and $\overline{\widehat{\boldsymbol{b}}}$ are completely imaginary with vector elements ω, $\widetilde{\omega}$ and $\widehat{\boldsymbol{b}}$ respectively. Consider that the delayed states $\boldsymbol{p_d}$ and $\overline{\boldsymbol{q_d}}$ are preserved.

The original orientation $\overline{\boldsymbol{q}}$ and its measure $\widehat{\overline{\boldsymbol{q}}}$ are altered by a slight rotation $\overline{\delta \boldsymbol{q}}$ executed in the local frame:

$$\overline{\boldsymbol{q}} = \widehat{\overline{\boldsymbol{q}}} \cdot \overline{\delta \boldsymbol{q}} \tag{4.3}$$

The rotation indicated by $\overline{\delta \boldsymbol{q}}$ is forced to be very small and can be estimated as

$$\overline{\delta \boldsymbol{q}} \approx \overline{\begin{bmatrix} 1 \\ \frac{1}{2} \delta \boldsymbol{\theta} \end{bmatrix}} \tag{4.4}$$

The rotation error $\delta \boldsymbol{\theta}$ is 3D state error. The state definition is not minimum because of the quaternion representation.

In the following equations, ${}^{c}\boldsymbol{F}$ and $\boldsymbol{G}$ represent the continuous state transition and noise input matrices correspondingly. The zeroth order estimation ${}^{d}\boldsymbol{F} = \boldsymbol{I} + {}^{c}\boldsymbol{F}.\ \Delta t$ gives the discrete time state transition model along which the state $\boldsymbol{x}_{k-1|k-1}$ and the covariance $\boldsymbol{P}_{k-1|k-1}$ can be propagated to the later time step k by means of equations:

$$\widehat{\boldsymbol{P}}_{k|k-1} = {}^{d}\boldsymbol{F} \widehat{\boldsymbol{P}}_{k-1|k-1} {}^{d}\boldsymbol{F}^{T} + \boldsymbol{G}^{d} \boldsymbol{Q} \boldsymbol{G}^{\boldsymbol{T}} \tag{4.5}$$

$$\widehat{\boldsymbol{x}}_{k|k-1} = \widehat{\boldsymbol{x}}_{k-1|k-1} + \Delta t \cdot \dot{\widehat{\boldsymbol{x}}}_{k-1|k-1} \tag{4.6}$$

Here ${}^{d}\boldsymbol{Q} = \Delta t \cdot {}^{c}\boldsymbol{Q}$ under the assumption that ${}^{C}\boldsymbol{Q}$ accommodates the noise densities $\eta_a, \eta_\omega, \eta_b$ and η_c on its crosswise.

All along the propagations, the postponed states and their collective covariances will be the same. On the other hand, the current state variables and covariances change due to the inertial measurements. The key point is that the covariances between the current state variables and the postponed states expand along propagation. They essentially indicate the observation about the corresponding transformation between these frames and grant us to combine frame-to-frame related calculations from the direct odometry system.

4.6.2 Measurement update

Between the reference and source stereo image pair, the measurement contains a 6-DoF corresponding pose calculation and its covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{6\times6}$. Since the position of the reference stereo image pair shows in the delayed states $\boldsymbol{p}_d$ and $\overline{\boldsymbol{q}_d}$, it is also possible to compute the measurement as an expression on the state $\boldsymbol{x}$. The translation $\boldsymbol{p}_\Delta$ and rotation $\overline{\boldsymbol{q}_\Delta}$ during the consecutive frames is measured in the source frame's universe. Declared as measurement residual we derive the measurement expression:

$$\boldsymbol{h}(\boldsymbol{x}) = \begin{bmatrix} \widetilde{\boldsymbol{p}_\Delta} - \boldsymbol{R}_{\overline{q_d}}^T(\widehat{\boldsymbol{p}} - \widehat{\boldsymbol{p}_d}) \\ \widetilde{\overline{\boldsymbol{q}_\Delta}}(\widehat{\overline{\boldsymbol{q}_d}}^{-1}\widehat{\overline{\boldsymbol{q}}}) \end{bmatrix} \tag{4.7}$$

Along with calculation of the Kalman gain matrix $\boldsymbol{K}$, the error state approximations are represented with $\boldsymbol{Kh}(x)$. Again, at this point, every state is estimated additively but for the quaternion. In consonance with eq. 4.3, the following is estimated again since $\overline{\boldsymbol{q}}_{k|k} = \overline{\boldsymbol{q}}_{k|k-1}\overline{\delta\boldsymbol{q}}$. $\overline{\boldsymbol{q}}_{k|k}$ desires to be normalized following this step to encounter the orientation quaternion has unit norm. At last, we reformulate the covariance matrix by means of:

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{KH})\boldsymbol{P}_{k|k-1}(\boldsymbol{I} - \boldsymbol{KH})^T + \boldsymbol{K\Sigma K^T} \tag{4.8}$$

Following the update step, the states $\boldsymbol{p}$ and $\overline{\boldsymbol{q}}$ are inserted into $\boldsymbol{p_d}$ and $\overline{\boldsymbol{q_d}}$ and their covariances are accepted outside of the initial states.

5. CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

Unmanned aerial vehicles (UAV) have increasing number of application areas, and autonomous solutions of UAVs are recently gaining more volume. Autonomous control of unmanned vehicles requires fast and accurate measurements of vehicle's pose and velocity in real-time. Although robust motion estimation can be achieved in outdoor UAV applications by utilizing GPS sensors, it is hard to achieve this robustness in low textured indoor environments. In this study, we use an active stereo-based visual odometry system to overcome motion estimation problem of indoor UAV applications. The proposed sensor system consists of a projector and stereo camera. The structured light source of sensor projects a pattern, which consists of lines with random colors, positions and orientations. The ambiguity of pattern feeds correspondence algorithm, thus created depth image is dense and accurate. Since motion has to be estimated fast for robustness of the overall system, we use GPU-accelerated image processing library, CUDA. Moreover, to overcome the shortage related with the feature-based odometry approach, we use direct approach.

The conventional visual odometry systems are based on tracking features in consecutive frames. For tracking those features, these systems employ feature description, extraction and matching algorithms. Therefore, computation load of the feature-based approaches is larger than the direct approaches, which is critical for real-time performance of the overall system. In order to increase the performance of visual odometry process, we use a direct method for motion estimation process. Without using any feature-based image-processing algorithm, the direct method calculates the motion by an iterative method until intensity error function converges or the termination criteria has been met. The key part of the direct methods is that they use the intensity values of all pixels, which are average of the pixel RGB color value. The motion estimation is determined when the pixel values in consecutive frames are equal. This direct approach increases the performance of the system on GPU dramatically. At the same time, this direct approach is capable of operating on a single core CPU.

Since indoor environments have sections like corridors, which have low-textured areas, passive stereo systems are not capable of supplying dense depth map in those areas. Therefore, we choose active stereo method to obtain dense disparity maps in typical indoor environments. In conditional active stereo application, uniform light patterns are employed. However, this homogeneity simplifies the correspondence problem, and results in uncertainty. We use heterogeneous line pattern to overcome simplification problem. To achieve this goal, we generate a line pattern consist of 200 lines which have random colors, orientations and positions.

In order to combine our sensor system with Dji Phantom, we integrate our sensor system with the controller Naza. Naza supplies IMU data to our system and communicate through PWM channels. We fuse IMU data with odometry data to increase the robustness of the system. Moreover, using IMU data is a backup precaution in case of a failure in odometry process.

In conclusion, this thesis indicates that it is possible to achieve robust motion estimation and obtain dense depth map of the environment with low textured areas. Moreover, using direct approach decrease the computation cost of the system, which shows that it is possible to implement these methods on a CPU computer. The system is implemented on a currently available quadrotor hardware and fuse with its control units.

5.2 Future Work

Possible improvements on visual odometry and suggestions on future works are given. Firstly, the depth map generated by stereo system does not always covers the whole image plane because of the points, which exist out of the camera range. For better performance, these irrelevant depth points can removed by using bucketing method. Limiting the depth range of the system can also increase the robustness of the motion estimation process.

Our structured light source equipped with its own lens, which limits the field of view of cameras. This limitation problem can be solved by using an extra lens. Our system also requires dark environment, since it uses visible light. Therefore, to operate in daylight, the projector and camera system can be replaced with infrared components.

Our visual odometry system implements a depth based error metric. Instead of depth based metric, a point-to-plane approach can be employed which is similar to iterative closest point (ICP).

In this study, the system uses all frames supplied from stereo camera. However, instead of aligning consecutive frames, the system can achieve better performance by aligning sufficiently closest frames. In this way, the drift of the motion estimation can be reduced for slow motions.

REFERENCES

[1] **Austin, R.** (2011). Unmanned aircraft systems: UAVS design, development and deployment (Vol. 54). John Wiley & Sons.

[2] **Nonami, K., Kendoul, F., Suzuki, S., Wang, W., & Nakazawa, D.** (2010). Autonomous Flying Robots: Unmanned Aerial Vehicles and Micro Aerial Vehicles. Springer Science & Business Media.

[3] **Makadia, A., & Daniilidis, K.** (2005, April). Correspondenceless ego-motion estimation using an imu. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Confer*ence on (pp. 3534-3539). IEEE.

[4] **Chonnaparamutt, W., & Kawasaki, H.** (2009, August). Fuzzy systems for slippage control of a pruning robot. In *Fuzzy Systems, 2009. FUZZ-IEEE 2009. IEEE International Conference* on (pp. 1270-1275). IEEE.

[5] **Zhang, J., & Singh, S.** Visual-Lidar Odometry and Mapping: Low-drift, Robust, and Fast.

[6] **Zindler, K., Geiß, N., Doll, K., & Heinlein, S.** (2014, October). Real-time ego-motion estimation using Lidar and a vehicle model based Extended Kalman Filter. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference* on (pp. 431-438). IEEE.

[7] **Bradski, G., & Kaehler, A.** (2008). Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.".

[8] **Harris, C. G., & Pike, J. M.** (1988). 3D positional integration from image sequences. *Image and Vision Computing*, 6(2), 87-90.

[9] **Shi, J., & Tomasi, C.** (1994, June). Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference* on (pp. 593-600). IEEE.

[10] Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006* (pp. 430-443). Springer Berlin Heidelberg.

[11] Lowe, D. G. (2004). Distinctive image features from scale-invariant key points. *International journal of computer vision*, 60(2), 91-110.

[12] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer vision–ECCV 2006* (pp. 404-417). Springer Berlin Heidelberg.

[13] Agrawal, M., Konolige, K., & Blas, M. R. (2008). Censure: Center surround extremas for real-time feature detection and matching. In *Computer Vision–ECCV 2008* (pp. 102-115). Springer Berlin Heidelberg.

[14] Gemeiner, P., Jojic, P., & Vincze, M. (2009, October). Selecting good corners for structure and motion recovery using a time-of-flight camera. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on* (pp. 5711-5716). IEEE.

[15] El-laithy, R. A., Huang, J., & Yeh, M. (2012, April). Study on the use of Microsoft Kinect for robotics applications. In *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION* (pp. 1280-1288). IEEE.

[16] Cruz, L., Lucio, D., & Velho, L. (2012, August). Kinect and rgbd images: Challenges and applications. In *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2012 25th SIBGRAPI Conference on* (pp. 36-49). IEEE.

[17] Deng, T., Li, H., Cai, J., Cham, T. J., & Fuchs, H. (2013, December). Kinect shadow detection and classification. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on* (pp. 708-713). IEEE.

[18] Shpunt, A., & Zalevsky, Z. (2013). *U.S. Patent No. 8,390,821*. Washington, DC: U.S. Patent and Trademark Office.

[19] Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

[20] Irani, M., & Anandan, P. (2000). About direct methods. In *Vision Algorithms: Theory and Practice* (pp. 267-277). Springer Berlin Heidelberg.

[21] Baker, S., & Matthews, I. (2004). Lucas-Kanade 20 years on: A unifying framework. *International journal of computer vision*, 56(3), 221-255.

[22] Mou, W., Wang, H., & Seet, G. (2014, June). Efficient visual odometry estimation using stereo camera. In *Control & Automation (ICCA), 11th IEEE International Conference* on (pp. 1399-1403). IEEE.

[23] Audras, C., Comport, A., Meilland, M., & Rives, P. (2011, December). Real-time dense appearance-based SLAM for RGB-D sensors. In *Australasian Conf.* on Robotics and Automation.

[24] Sturm, J., Engelhard, N., Endres, F., Burgard, W., & Cremers, D. (2012, October). A benchmark for the evaluation of RGB-D SLAM systems. *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference* on (pp. 573-580). IEEE.

[25] Stowers, J., Hayes, M., & Bainbridge-Smith, A. (2011, April). Altitude control of a quadrotor helicopter using depth map from Microsoft Kinect sensor. In *Mechatronics (ICM), 2011 IEEE International Conference* on (pp. 358-362). IEEE.

[26] Steinbrucker, Frank, Jürgen Sturm, and Daniel Cremers. "Real-time visual odometry from dense RGB-D images." *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference* on. IEEE, 2011.

[27] Whelan, T., Johannsson, H., Kaess, M., Leonard, J. J., & McDonald, J. (2013, May). Robust real-time visual odometry for dense RGB-D mapping. In *Robotics and Automation (ICRA), 2013 IEEE International Conference* on (pp. 5724-5731). IEEE

[28] Stückler, J., & Behnke, S. (2012, May). Robust real-time registration of RGB-D images using multi-resolution surfel representations. In *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference* on (pp. 1-4). VDE.

[29] Engelhard, N., Endres, F., Hess, J., Sturm, J., & Burgard, W. (2011, April). Real-time 3D visual SLAM with a hand-held RGB-D camera.

[30] Scharstein, D., & Szeliski, R. (2003, June). High-accuracy stereo depth maps using structured light. In *Computer Vision and Pattern Recognition, 2003.*

[31] Newcombe, R. A., Lovegrove, S. J., & Davison, A. J. (2011, November). DTAM: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on* (pp. 2320-2327). IEEE.

[32] Scaramuzza, D., & Fraundorfer, F. (2011). Visual odometry [tutorial]. *Robotics & Automation Magazine, IEEE*, *18*(4), 80-92.

[33] Irani, M., & Anandan, P. (2000). About direct methods. In *Vision Algorithms: Theory and Practice* (pp. 267-277). Springer Berlin Heidelberg.

[34] Url – 1 <*http://eigen.tuxfamily.org/*>, date retrieved 23.04.2015.

[35] Url – 2 <http://www.boost.org/>, date retrieved 23.04.2015.

[36] Baker, S. (2014). Inverse Compositional Algorithm. In *Computer Vision* (pp. 426-428). Springer US.

[37] Sturm, J., Engelhard, N., Endres, F., Burgard, W., & Cremers, D. (2012, October). A benchmark for the evaluation of RGB-D SLAM systems. In*Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on* (pp. 573-580). IEEE.

[38] Grewal, M. S., & Andrews, A. P. (2014). *Kalman filtering: Theory and Practice with MATLAB*. John Wiley & Sons.

APPENDICES

APPENDIX A : Representing Rotations With Quaternions

Quaternions are a development to the complex number practice where a quaternion is defined by one real and three imaginary components. They establish a significant appearance in mathematics, computer vision and computer graphics.

A Quaternion is shown as:

$$\tilde{q} = q_s + q_a i + q_b j + q_c k \tag{A.1}$$

where q_s, q_a, q_b and q_c are real number and i, j and k are defined as:

$$\begin{aligned} i^2 &= -1, ij = -ji = k \\ j^2 &= -1, jk = -jk = i \\ k^2 &= -1, ki = -ik = j \end{aligned} \tag{A.2}$$

For comfort of symbolization, we utilize the below statements vice versa:

$$q_s + q_a i + q_b j + q_c k = \tilde{q} \Leftrightarrow \vec{q} = \begin{bmatrix} q_s \\ q_a \\ q_b \\ q_c \end{bmatrix} \tag{A.3}$$

For two quaternions $\tilde{q}_1 = q_{1,s} + q_{1,a} i + q_{1,b} j + q_{1,c} k$ and $\tilde{q}_2 = q_{2,s} + q_{2,a} i + q_{2,b} j + q_{2,c} k$, addition and subtraction of two quaternions are easy to understand.

$$\begin{aligned} \tilde{q}_1 + \tilde{q}_2 = (q_{1,s} + q_{2,s}) + (q_{1,a} + q_{2,a})i + (q_{1,b} + q_{2,b})j \\ + (q_{1,c} + q_{2,c})k \end{aligned} \tag{A.4}$$

$$\begin{aligned} \tilde{q}_1 - \tilde{q}_2 = (q_{1,s} - q_{2,s}) + (q_{1,a} - q_{2,a})i + (q_{1,b} - q_{2,b})j \\ + (q_{1,c} - q_{2,c})k \end{aligned} \tag{A.5}$$

Multiplying of two quaternions is explained as their Hamiltonian product. Accomplishing adopt of the multiplication guidelines for three imaginary items, the multiplication is shown as:

$$\tilde{q}_3 = \tilde{q}_1\tilde{q}_2 \Rightarrow \begin{matrix} s_3 = q_{1,s}q_{2,s} - q_{1,a}q_{2,a} - q_{1,b}q_{2,b} - q_{1,c}q_{2,c} \\ a_3 = q_{1,s}q_{2,a} + q_{1,a}q_{2,s} + q_{1,b}q_{2,c} - q_{1,c}q_{2,b} \\ s_3 = q_{1,s}q_{2,b} - q_{1,a}q_{2,c} + q_{1,b}q_{2,s} + q_{1,c}q_{2,a} \\ s_3 = q_{1,s}q_{2,c} - q_{1,a}q_{2,b} - q_{1,b}q_{2,a} + q_{1,c}q_{2,s} \end{matrix} \tag{A.6}$$

Norm of a quaternion is can be shown as:

$$\|\tilde{q}\| = \sqrt{q_s^2 + q_a^2 + q_b^2 + q_c^2} \tag{A.7}$$

Inverse of a quaternion is calculated as:

$$\tilde{q} * \tilde{q}^{-1} = 1 \Rightarrow \tilde{q}^{-1} = \frac{q_s - q_a i - q_b j - q_c k}{\|q\|} \tag{A.8}$$

Two appropriate conditions of quaternions that fall in interest in scope of this study are unit quaternion, formulized as a quaternion with unit norm, and pure quaternion, formulized as a quaternion with zero real part.

Note that the inverse of a unit quaternion is identical to the complex conjugate of that quaternion, $\tilde{q}^{-1} = \tilde{q}^*$.

Unit quaternions can be utilized to show rotations about a random axis. For a point 3D point p with coordinates p_x, p_y and p_z, when we determine an axis of rotation as a unit pure quaternion $\tilde{u} = x_u i + y_u j + z_u k$ and magnitude of rotation in radians as θ, the related quaternion showing that rotation is described as $\tilde{q} = \cos\left(\frac{\theta}{2}\right) + \sin(\frac{\theta}{2})u$. After that the rotation can be defined as stated in to Hamilton-Cayley formula as:

$$\tilde{p}' = \tilde{q}\tilde{P}\tilde{q}^{-1} \tag{A.9}$$

Where $\tilde{P}$ is described as the pure quaternion showing the point p, $\tilde{P} = p_x i + p_y j + p_z k$. $\tilde{P}'$ is the pure quaternion showing the rotated point, p'.

APPENDIX B : Quaternions and Matrices

A rotation in 3D can be shown by an orthogonal 3×3 matrix R with unit determinant. A rotation quaternion can be altered to a rotation matrix by characterizing the Hamilton-Cayley formula with a matrix. The consequence descripting is appointed the Euler-Rodrigues formula.

$$\begin{matrix} \tilde{p}' = \tilde{q}\tilde{P}\tilde{q}^{-1} \\ p' = Rp \end{matrix} \Rightarrow R$$
$$= \begin{bmatrix} q_s^2 + q_a^2 - q_b^2 - q_c^2 & -2q_sq_c + 2q_aq_b & 2q_sq_b + 2q_aq_c \\ 2q_sq_c + 2q_aq_b & q_s^2 - q_a^2 + q_b^2 - q_c^2 & -2q_sq_a + 2q_bq_c \\ -2q_sq_b + 2q_aq_c & 2q_sq_a & q_s^2 - q_a^2 - q_b^2 + q_c^2 \end{bmatrix} \tag{B.1}$$

For the inverse transformation, we can achieve

$$\begin{aligned} q_s^2 &= (1 + r_{11} + r_{22} + r_{33})/4 \\ q_a^2 &= (1 + r_{11} - r_{22} - r_{33})/4 \\ q_b^2 &= (1 - r_{11} + r_{22} - r_{33})/4 \\ q_c^2 &= (1 - r_{11} - r_{22} + r_{33})/4 \end{aligned} \tag{B.2}$$

$$\begin{aligned} q_sq_a &= (r_{32} - r_{23})/4 \\ q_sq_b &= (r_{13} - r_{31})/4 \\ q_sq_c &= (r_{21} - r_{12})/4 \\ q_aq_c &= (r_{13} - r_{31})/4 \\ q_aq_b &= (r_{21} - r_{12})/4 \\ q_bq_c &= (r_{32} - r_{23})/4 \end{aligned} \tag{B.3}$$

Consider the signs of the quaternion parameters are puzzling. We can choose a sign for one of the four elements. In this study, for numerical establishment, firstly, all parameters are calculated utilizing given equations. After all, the largest one preferred and the others are recalculated.

APPENDIX C : Specifications of Sensor Components

We have use three on the shelf product for control of quadrotor and process visual odometry software. Specifications of the products are given in below tables.

Table C.1 : Jetson TK1

Specification	**Feature**
Model	Jetson TK1 Development Board
SOC	Kepler GPU with 192 CUDA Cores & 4-Plus-1quad-core ARM Cortex A15 CPU
Memory	2GB × 16 memory with 64 bit width 16GB 4.51 eMMC memory
Mini Card Slots	1 × Half mini-PCIE
SATA	1× SATA
Rear Panel Ports	1 × RS232 serial port 1 × HDMI 1 × USB 3.0
Other Connectors	Full size SD/MMC connector JTAG DP/LVDS Touch SPI GPIO UART HSIC I2c

Table C.2 : LI-USB30-V024STEREO

Specification	**Feature**
Connection	USB 3.0 Super Speed Support
Output	Raw data output without compression
Camera	Dual Camera
Size	80mm × 15mm
Mass	12g
Video Resolution @Frame Rate	640× 480 @ 0 fps
Sensor	Aptina MT9V024 Global Shutter WVGA Sensor

Table C.3 : Dji Phantom V1.2

Specification	**Feature**
Power Consumption	3.12W
Take-Off Weight	<1200g
Max Yaw Angular Velocity	$200°/s$
Max Tilt Angle	45°
Max Flight Velocity	10m/s
Diagonal Distance	350mm

CURRICULUM VITAE

Name Surname: Bulut Kırca

Birth Place and Date: İstanbul, 17.10.1989

E-mail: bulutkirca@gmail.com

Bachelor Degree: İstanbul Technical University Electrical-Electronics Engineering Department Electrical Engineering Programme (2007 – 2012)

Work Experience: EV Elektrikli Araç Sistemleri A.Ş. (2012-2014)

Publications:

- **Kırca B.**, Altuğ E., "İnsansız Araçlar İçin Piksel Yoğunluğu Takibi ile Gerçek Zamanlı Doğrudan Görsel Poz Kestirimi", *TOK'2015 Otomatik Kontrol Türk Milli Komitesi Ulusal Toplantısı*.
- Avcı G., Sarıoğlu B., **Kırca B.**, Aryan, A. J., Karakaş S., Uz G. "Introduction and Concept Design of the 'AUVTECH' *Autonomous Underwater Vehicle", AUVSI & ONR's 15th RoboSub Competition*, 2012.