אוניברסיטת בן-גוריון בנגב

Ben-Gurion University of the Negev

הפקולטה למדעי ההנדסה

המחלקה להנדסת חשמל ומחשבים

Faculty of Engineering Science

Dept. of Electrical and Computer Engineering

פרויקט הנדסי שנה ד'

Fourth Year Engineering Project

דו״ח התקדמות

Progress Report

טיסה וניווט אוטונומיים של רחפנים ללא מערכת איכון גלובלית

Autonomous flight and navigation of UAV without GPS

Project number:	**p-2016-001**		**מספר הפרויקט:**
Students (Name & ID):	Asaf Sarid 301465258 Ohad Cohen 200654606	301465258 אסף שריד 200654606 אוהד כהן	**סטודנטים (שם ו ת.ז.):**
Supervisors:	Prof. Guterman Hugo Zohar Ilan	פרופ' הוגו גוטרמן אילן זוהר	**מנחים:**
Sponsors:			**תומכים:**
Submitting date:	17/1/2016		**תאריך הגשה:**

Table of Contents

1. Introduction

Our project goal is to allow UAV to navigate from a known starting point to a given target print and avoiding from obstacles, by using optical flow technique and distance sensors. We have already implemented flight controller, which receive the desired coordinates and fly the UAV to this target point. The feedback of the measured location and angles are now come from analyzing the physics of the UAV, but in real world this is not enough since we have mechanic deviation and drift error.

In addition, we started to design and implement optical flow algorithm. This algorithm purpose is to point the current location as feedback to the flight controller, using an on board installed camera. Our algorithm includes calibration, and can handle pitch and roll in the camera angles.

Our next steps are:

- Fully integrate the simulated controller (written in Matlab and Simulink) with the optical flow algorithm (written in OpenCV- C++).
- Perform tests and simulate more complex scenarios to make sure that our controller can handle all of them.
- Convert the simulator to real-time on-chip controller, and start tests with the UAV (changing and upgrading the controller during this step).
- Add avoiding obstacles feature algorithm.

2. Flight Controller[1]

After deep literature survey, we implemented a flight controller.

We started by implementing only the attitude controller, and tried to insert it to Simulink framework of existing project, but we ran into some difficulties. We decided to start from scratch.

The controller is compound from 3 main blocks - Position Controller, Attitude Controller and Physics:

- The Position Controller (PID) control the location (X, Y) of the UAV, meaning it gets the desired location of the UAV with the measured Euler angles and translate it into the command Euler angles.
- The Attitude Controller get the output of the Position Controller (desired Euler angles) and the desired Z, and all the measured parameters (Location, velocity and angles) and output the moments and torques that required from the motors. (This data is manipulated to become output to motors- through H sub block).
- The Physics block gets the motors RPM and outputs the measured information: Location, Position, Angles and Angular velocity.

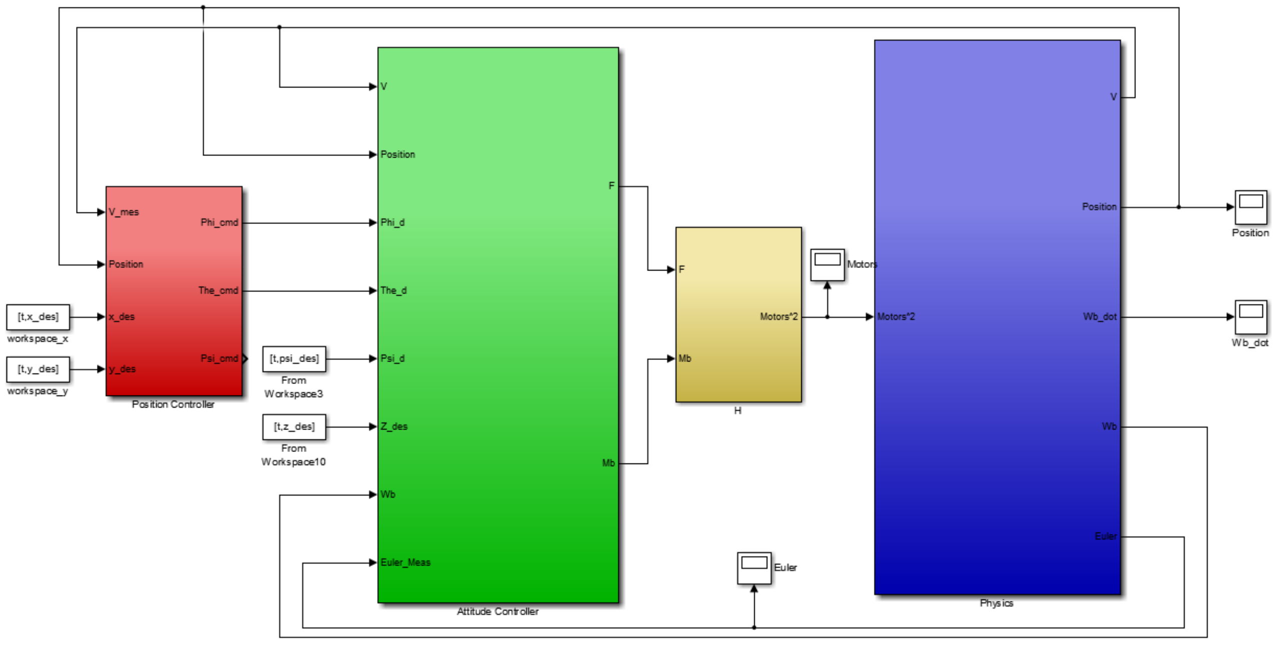


Figure 1- Top Level Flight Controller

The controller first version is theoretical; hence the results will not be similar to reality. We will tune the PID constants again later. In addition, we will add the feedback from the optical flow algorithm, so there will not be accumulated error from the IMU.
We can give the controller the desired location (X and Y) and altitude (Z), and the controller will take the UAV to this location.
Results of some tests are in Appendix 1.

3. Optical Flow

3.1. General Explanation

The optical flow technique is very popular nowadays in Computer Vision (for tracking moving objects for example). We are currently using one of these technique algorithms: Farneback.
In the next few weeks we will evaluate more methods of estimation using optical flow, so we may replace current method (Farneback) with more accurate and fit to our needs method.

Processing the input frames from the camera using this algorithm gives us the current location of the UAV (without using the IMU). In addition, we need to handle two initial issues: calibrate the camera, and add transformation functionality for states when the camera is not aligned with the surface (UAV pitch and roll angles). We will discuss those issues in the next sections.

Our algorithm reset the location on initialization, and accumulates it, so every frame we can know the current location of the UAV. We captured few videos, and applied our algorithm with them as input. Our goal was to find the optimal parameters- considering accuracy and time of processing. The results are in the appendixes [Appendix 2].
The output from the optical code function we integrate into the controller as the feedback to the PID controller (Currently this feedback comes from the Physics block in the controller- but this feedback is theoretical and not taking into consideration the mechanical deviation, drift error etc).

3.2. The Farneback Algorithm[2][3]

Our algorithm is based on the Farneback algorithm, which is located in the OpenCV built-in libraries. The Farneback algorithm fits to our requirements because it is using fixed grid (according to pre-defined parameters), we use this data to calculate the distance from a frame to the next frame- and to conclude the velocity and location of the UAV.
At first we tried other built-in algorithms, but most of them locate spots and finds the location of them in the next frame. Those algorithms are better for static camera that needs to follow moving objects, but is not accurate enough for us. The method is described in Gunner Farneback paper. The method has two steps- The first step is to approximate each neighborhood of both frames by quadratic polynomials, which can be done efficiently using the polynomial expansion transform. From observing how an exact polynomial transforms under translation a method to estimate displacement fields from the polynomial expansion coefficients is derived and after a series of refinements leads to a robust algorithm. Evaluation shows good results [6].

3.3. Calibration of the camera

Geometric Camera Calibration (or Camera Resectioning)[4] is the process of estimating the parameters of a pinhole camera model approximating the camera that produced a given photograph or video. Usually, the pinhole camera parameters are represented in a 3 × 4 matrix called the camera matrix, divided into intrinsic and extrinsic parameters. The extrinsic parameters are used in transformation from World coordinates to Camera coordinates, while the intrinsic are used in transformation from Camera coordinates to Pixel Coordinates.
These parameters refer to the pinhole camera model- a theoretical ideal model. But the camera has also a lens, causes radial and tangential distortion. This distortion can be compensate using this equations [5]:

$$X_{distorted} = x + [2p_1xy + p_2(r^2 + 2x^2)]$$

$$Y_{distorted} = y + [p_1(r^2 + 2y^2) + 2p_2xy]$$

$$p_1 and\ p_2 - Tangential\ distortion\ coefficients\ of\ the\ lens$$

In our project we need to calibrate the camera that we are using in order to improve the accuracy of the measurements of the location, for now we did it using OpenCV built in functions and template of chess board.

3.4. From Camera coordinates to World coordinates

Our camera is installed on the UAV, and the angles of the camera changing according the UAV. In order to know our location using the camera, we need to transform the original captures frames from the UAV plane to a plane that is parallel to the surface (Pitch and Roll angles or zero). Assuming we are not doing so, the calculation will be less accurate. For example, the pixels that are closer to the image center (bottom of the frame) appear to move slower and the more far pixels appear to move faster, while after the transformation we will see that all the pixels are moving at the same rate.
Our current processing of the frame finds the average velocity in X and Y axis of all the pixels, so we must transform the plane of the capture.
We will use rotation matrix to handle this situation, the Pitch and Roll angels will be taken from the internal built in IMU of the UAV.

4. Conclusions

The integration of the controller and the feedback using optical flow is not easy. The OpenCV platform gives the option to get the distance that a pixel in the frame has traveled, hence it is very important to clean all the noise and to select the optical flow algorithm that best fits our usage.
In addition, one of the most challenging aspects of the project is to transform the controller and the optical flow algorithm into one platform- Matlab/Simulink or OpenCV. The first will allow us to manipulate and test the controller during the development of the project, and to collect and compare result using graphs. The second, on the other hand, can be burned into the hardware of the UAV (and the optical flow is built in into this platform). In order to implement accurate and efficient controller we must run many scenarios, and to see if the simulated UAV behaves as expected. When we will start using the controller on real UAV, we will make another fine tuning to the constants and the parameters.

Another issue we must take care of is the special condition we have in our project. For example, we may have GPS communication once in a while- this can reset our error. But at the same time, we may suffer from strong wind because the UAV flying outdoor- and this can increase the drift error over time. We need to adjust the final controller to these conditions.

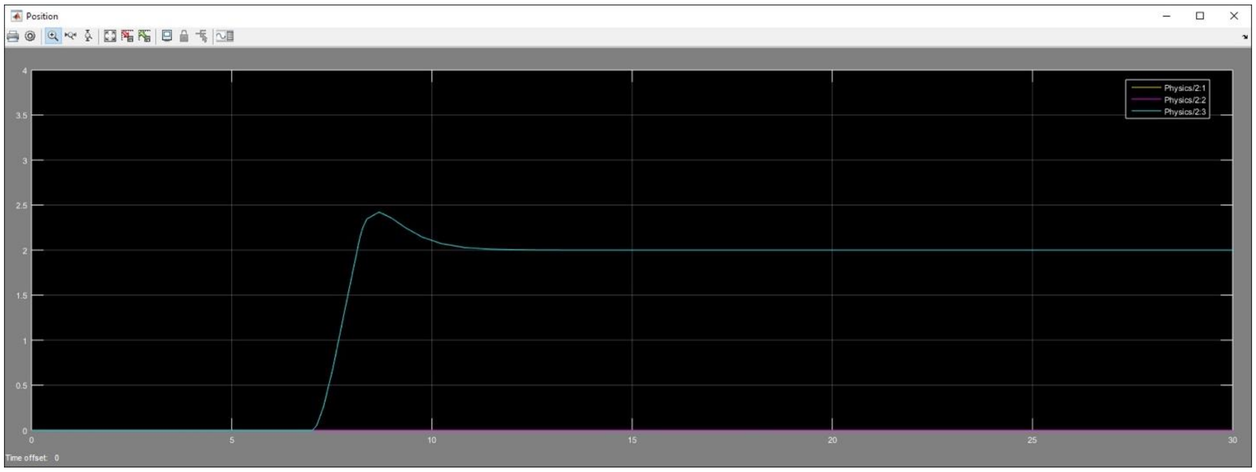
5. References

[1] C. Balas, "Modelling and Linear Control of a Quadrotor", 2007.
[2] Optic Flow. Scholarpedia. Available: http://www.scholarpedia.org/article/Optic_flow#A3_Method_of_Farneb.C3.A4ck_.282000.29 Accessed Jan. 14, 2016.
[3] Dense Motion Estimation. SCRIBDS. Available: https://www.scribd.com/fullscreen/210750997?access_key=key-2a6845xnkhkk8naxnxfq&allow_share=true&escape=false&view_mode=scroll. Accessed Jan. 16, 2016.
[4] Camera Resectioning. Wikipedia. Available: https://en.wikipedia.org/wiki/Camera_resectioning. Accessed Jan. 12, 2016.
[5] Camera Calibration. *Mathworks*. Available: http://www.mathworks.com/help/vision/ug/camera-calibration.html. Accessed Jan. 16, 2016.
[6] G. Farneback, "Two-Frame Motion Estimation Based on Polynomial Expansion", 2004

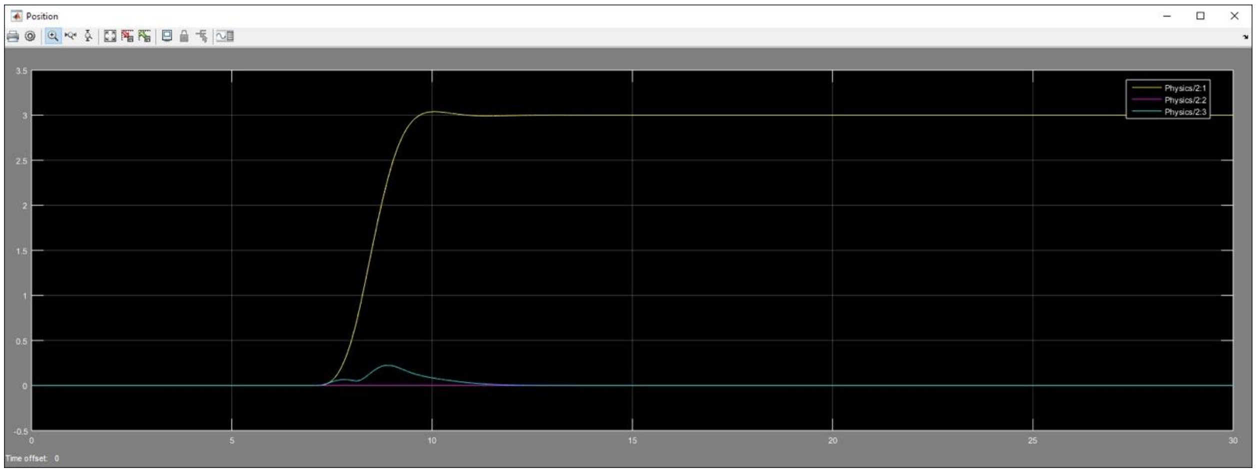
6. Appendix

6.1. Flight controller tests:

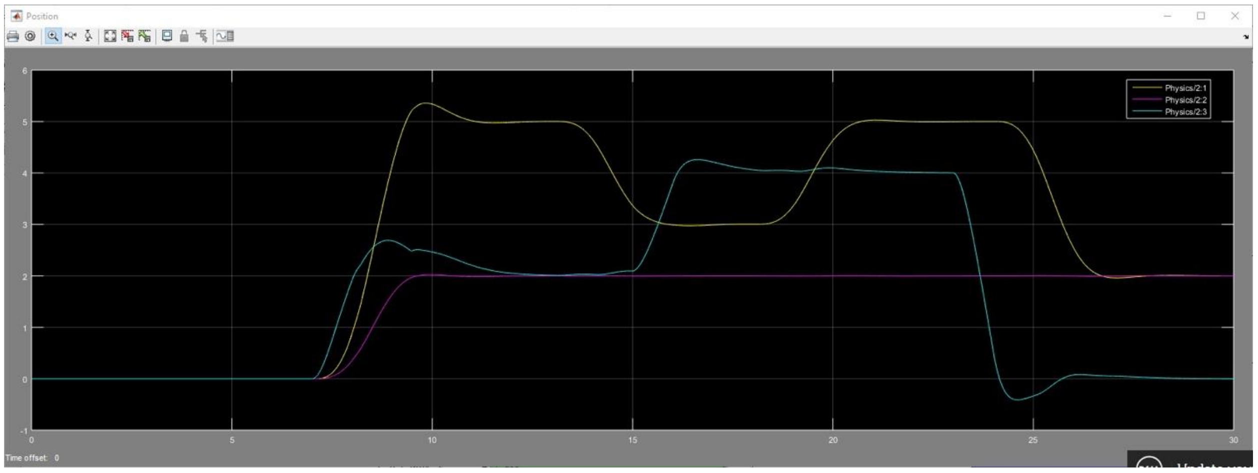
Z axis only: $Z = \begin{cases} 0 & t < 8 \\ 2 & t \geq 8 \end{cases}$



X axis only: $X = \begin{cases} 0 & t < 8 \\ 3 & t \geq 8 \end{cases}$

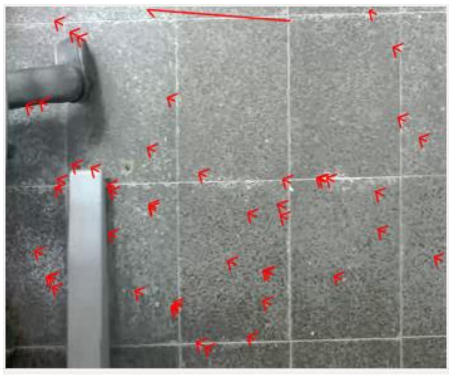


All axes changing: $Z = \begin{cases} 0 & t < 8 \\ 2 & 8 \leq t < 16 \\ 4 & 16 \leq t < 24 \\ 0 & 24 \leq t \leq 30 \end{cases}$ $Y = \begin{cases} 0 & t < 8 \\ 2 & t \geq 8 \end{cases}$ $X = \begin{cases} 0 & t < 8 \\ 5 & 8 \leq t < 14 \\ 3 & 14 \leq t < 19 \\ 5 & 19 \leq t < 25 \\ 2 & 25 \leq t \end{cases}$

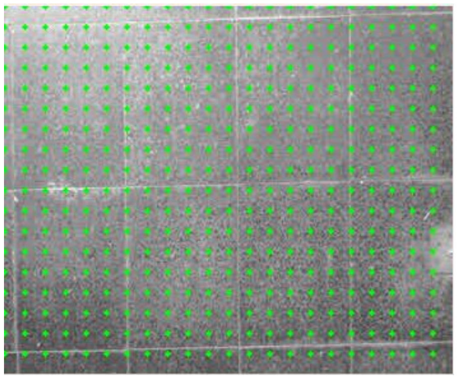


6.2. Optical flow experiments

Surface tracking using Lucas Kanade method (previously used method)



Surface tracking using Farneback method (currently used method)



6.3. Gantt Table

Start Date	End Date	Cohen Ohad	Sarid Asaf	Task	Status	Adviser Approved	Adviser Notes
25/10/2015	20/11/2015	yes	yes	Basics: Rigid Body, QuadCopter, etc.	Done	no	
26/10/2015	20/11/2015	yes	yes	Literature Survey	Done	no	
26/10/2015	27/11/2015	yes	yes	Simulator: Installation and Activation	Done	no	
30/11/2015	01/01/2016	no	yes	Simulator: Deeper Understanding and Implementation	Done	no	
30/11/2015	28/12/2015	yes	no	Designing and Implementing the controller (Selecting Controller Method)	In Progress	no	
29/12/2015	15/01/2016	yes	yes	Testing the controller on the UAV	In Progress	no	
18/01/2016	04/02/2016	yes	yes	Learning Optical Flow navigation technique	In Progress	no	
5/02/2016	04/03/2016	no	yes	Designing navigation algorithm	Not Done	no	
7/03/2016	04/04/2016	yes	no	Designing system for identifying and avoiding obstables	Not Done	no	
5/04/2016	12/05/2016	yes	yes	Integration of the systems	Not Done	no	

6.4. Grading

המלצת ציון לדו"ח מכין

אם יש צורך, לכל סטודנט/ית בנפרד

מספר הפרויקט: P-20____-_____

שם הפרויקט:

שם המנחה החיצוני:

שם המנחה מהמחלקה:

שם הסטודנט/ית: ת.ז.:

%		חלש 64-55	בינוני 65-74	טוב 84-75	ט"מ 94-85	מצוין 100-95	
25	הערכת ההתקדמות בעבודה בפועל בהשוואה לתכנית העבודה						
25	מציאת פתרונות לבעיות שהתגלו ויישומם						
20	גילוי יוזמה וחריצות						
30	מקוריות ותרומה אישית						

הערות: