# Integration of a Vicon camera system for indoor flight of a Parrot AR Drone

**Conference Paper** · December 2015

DOI: 10.1109/ISMA.2015.7373476

CITATIONS
4

READS
2,028

**2 authors:**

Ahmed Mashood
United Arab Emirates University
**5** PUBLICATIONS **28** CITATIONS
SEE PROFILE

Hassan Noura
Aix-Marseille Université
**171** PUBLICATIONS **1,879** CITATIONS
SEE PROFILE

# Integration of a Vicon Camera System for Indoor Flight of a Parrot AR Drone

Shaima Al Habsi, Maha Shehada,
Marwah Abdoon
Department of Electrical Engineering
UAE University
Al Ain, Abu Dhabi, UAE
e-mail: {200935035, 201050385, 201004889}@uaeu.ac.ae

Ahmed Mashood, Hassan Noura
Department of Electrical Engineering
UAE University
Al Ain, Abu Dhabi, UAE
e-mail: {a.mashood, hnoura}@uaeu.ac.ae

*Abstract*— **The main objective of this project is to design and implement an indoor autonomous flight control for Unmanned Aerial Vehicle (UAV) of type Parrot AR Drone. Autonomous flight is achieved when the AR Drone is capable of following a predefined trajectory without any human interaction. For this purpose, a control algorithm for Parrot AR Drone designed by Mathworks under software Matlab\Simulink was used and enhanced. As a part of the control model development and enhancement, a Vicon Capture System consisting of a set of cameras was integrated in real time with the AR Drone Matlab\Simulink. Vicon Capture System was used to capture the position of AR Drone and compare it with the estimated position obtained from AR Drone on-board sensors. It was noticed that the Vicon cameras' position estimation was closer to the real position than the sensors' estimation.**

***Keywords—Simulink development kit; Vicon capturing system; Autonomous flight; Parrot ARDrone2.0***

## I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have become an essential element in automation and a notable mark in all aspects of technology nowadays. It was thought previously that UAVs appear only in research, education and military applications, but UAVs have recently proved that it can be a part of almost any application in our lives starting by telecommunication, agriculture or medicine, and reaching food delivery. UAVs have become so popular due to its light weight and convenience of use. Many applications and software were developed to ease using UAVs to all types of users and also to interface the UAVs to other equipment and enhance their performance. In addition, many researches were done to develop the functionality of the UAVs and more importantly to make them completely automated, which means having UAVs programmed to fly, get data, analyze it, make conclusions and take decisions without any human interaction.

Engel, J. (2011) [1] has worked on flying and controlling an AR Drone autonomously in an unknown environment utilizing the onboard sensors only. Mainly, the position estimation was based on the visual analysis of what is seen by the front camera to estimate the pose of the drone and thus generating the required control commands which keeps the drone going on the desired path in three-dimensional space. The measurements obtained by the onboard inertial measurement unit (IMU) - which consists of an accelerometer and a gyroscope- were used to compensate for the temporary loss of visual tracking. Another difference between the work presented here and the work done by Engel, was Engel using the software development kit (SDK) provided by Parrot, while the work here is developed by an SDK which is MATLAB / Simulink based.

While Engel has used the onboard front camera, Yue, S. (2012) [2] has used an external low-resolution camera sensor, namely Kinect, to find the approximate position of the AR Drone during indoor flight. The position was given as feedback to the drone model which is derived from Kinemics and Dynamics of common quadrotors. The model was simulated using both MATLAB and C++. A PID controller was used as the main controller and has been enhanced with various filters designs, such as low/high pass filter, Complementary Filter and Kalman Filter.

AR Drone hasn't been the only platform used in UAVs' field of research. Aerial Network Guided Electronic Lookout (ANGEL) was the platform used by Schmidt, M. (2011) [3] who has limited his research to the simulation and control of this quadrotor in the near-hover position using MATLAB / Simulink. The position estimation in Schmidt's work was done by processing the measurements of one onboard IMU which itself consists of a dual-axis gyroscope and a triple-axis accelerometer.

Some of the above research has gone more into UAV modeling and simulation than the autonomous flight, while for the work presented in this paper autonomous flight has been the focus. The main objective was to enhance the performance of a Parrot AR Drone 2.0 such that it flies autonomously on a predefined path using its onboard sensors' measurements, and studying the possibility of using the Vicon cameras' measurements and position estimation to enhance the performance of the UAV. Next section introduces Parrot AR Drone 2.0 which is the UAV used in this work. It also describes the general mathematical model which represents the dynamics of the UAV. After that, a very useful MATLAB / Simulink tool which is used to fly the drone autonomously is described along with analyzing the results obtained using this tool. After that, a section introduces the Vicon camera that has been studied as an option to improve the position estimation of the drone. Following that, a section shows the comparison and the combination of data obtained by the MATLAB / Simulink and the Vicon Camera System. Finally, last section presents conclusions and problems faced in addition to giving some recommendations.

## II. Parrot AR Drone 2.0

The Parrot AR Drone 2.0 power edition is one type of UAV where AR stands for Augmented Reality. It is considered to be a multi-copter UAV or basically a quadrotor UAV. Parrot AR. Drone 2.0 was built by a French company called Parrot. Parrot AR. Drone 2.0 is typically composed of four propellers with four electrical motors, one cross beam, two cameras, two electrical boards, base house which is the place in where all the above mentioned components are connected together, two top covers (indoor and outdoor hulls) used to protect the UAV [2]. Figure 1 shows Parrot AR Drone 2.0 power edition with indoor hull while Figure 2 shows the drone with outdoor hull.

Figure 1. AR.Drone 2.0 with Indoor Hull

Figure 2. AR Drone 2.0 with Outdoor Hull

There are four different types of sensors available in AR Drone 2.0 which are accelerometer, 3-axis gyroscope, magnetometer, and ultrasound altimeter sensor.

The Parrot AR Drone 2.0 moves based on six degrees of freedom (DOF) which are the translational movement along the X, Y and Z axis and the rotational movements around the Pitch, Yaw and Roll angles. The drone has four basic movements which are roll, yaw, throttle and pitch movements.

Many mathematical models have been developed for the quadcopters in the literature. One of the mathematical model presented by Yue thesis [2] describes the drone behavior in a logical and simple steps as follows.
First of all rotational and translational vectors were written to represent the position of the drone with respect to earth frame:

$$S^E = [x\ y\ z]^T \quad , \quad \Theta^E = [\phi\ \theta\ \psi]^T$$

Then those vectors were converted to body frame since force and torque are affecting the body itself, thus it is meaningless to look at the earth frame:

$$V^B = [u\ v\ \omega]^T \quad , \quad \Omega^B = [p\ q\ r]^T$$

Where V (translational speed) is the derivative of S, while Ω (rotational speed) is the derivative of Θ.

Afterward, the relation between translational position $S^E$ in earth frame and linear velocity $V^B$ in body frame was given as:

$$\dot{S}^E = V^E = R_\theta V^B$$

Where $R_\theta$ is the rotational matrix. In a similar way the rotational position $\Theta^E$ in earth frame can be related to the angular velocity $\Omega^B$ in body frame using the translational matrix $T_\theta$:

$$\Theta^E = \Omega^E = T_\theta \Omega^B$$

## III. Software Development Kit of AR.Drone 2.0 (SDK)

The first objective of this work was to control one drone autonomously to be following a predefined path. As a starting point it was very useful to see what was accomplished by previous researchers and software developers. One of the good tools which was found on the Mathworks website is a Simulink model called AR Drone Software development kit v1.1(SDK) which was developed by David Sanabria [4].

This Simulink model is consisting of many blocks which are integrated together to control the drone such that it follows a path that has been defined by the user in the form of waypoint coordinates. The development kit actually contains two program. One is used for Wi-Fi control which takes the real sensors' measurements from the drone, compare them with the desired waypoints given by the user and calculate the required control inputs which are sent as commands to the drone to control it. This model requires having AR Drone connected to the PC via Wi-Fi during execution.

The other program is used for simulation which differs of the previous model in one block. In the Wi-Fi control model there is a block for obtaining the sensors measurements from the drone, while in this model that block is replaced by a block which includes all mathematical equations representing the behavior or the model of the drone. The sensor values do not exist here, rather, the feedback from the drone is estimated using its mathematical model.

The Wi-Fi control model does not require any additional sensors to be used. It basically uses the information obtained by the existing sensors on the drone, which measure the drone acceleration in three directions (measured by three accelerometers), and the three angles' rates (measured by three gyroscopes). The acceleration and angle rate values are integrated and processed to estimate the position of the drone, then the estimated position is compared with the desired points which are given by the user and finally commands are sent to the drone in the form of the three angle rates and the vertical velocity via Wi-Fi connection made between the computer and the drone.

The main block for Simulink model (Figure 3) sends all commands to the drone and then it gets back the measurements of the sensors. The inputs of this block are the estimated references or set points sent to the drone as commands via Wi-Fi, those values are calculated in another block (Baseline Controller) as will be explained shortly.

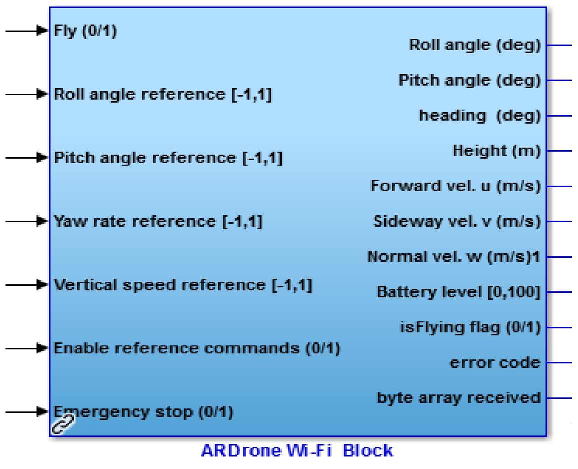

Figure 3.AR Drone Wi-Fi Block

After sending the commands to the drone; the main block has "packet input" communication block to receive all the navigation information from the drone. The information comes all in one packet so the AR.Drone Data Decoding block breaks down this packet and extracts all pieces of information from it. The information obtained are battery level, roll angle, pitch angle, height, forward speed, sideway speed and normal speed. This indicates that there is an integrator already on the drone which does the first integration to convert angles rates and acceleration to angles and velocities. The values of angles, velocities and height come out of the Wi-Fi block going toward the green block which is responsible for position estimation (Figure 4).

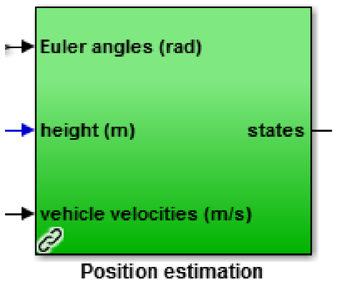

Figure 4.Position Estimation Block

The yaw angle is used in addition to the forward and side velocities to estimate the position by decomposing the velocities to its x and y axis components, adding up x components and y components separately and then integrating each axis velocity to find the position of the drone with respect to that axis as shown in Figure 5.

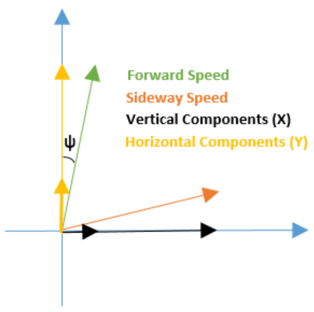

Figure 5.Velocities Decomposition Using Yaw Angle

It is notable here that the origin of these x-y axis (0, 0 coordinate) are assumed to be the point from which the drone has taken off, so each time it takes off sensors are calibrated and that point is considered as (0, 0). Having x and y position estimated and knowing the height makes the position in x-y-z space known, all these information (angles, velocities, position in x-y plane and height) are combined in one variable called states and forwarded to the next block (baseline controller, Figure 6), where these values are compared with the reference points given by the user at the beginning. As the scope of this paper is limited to the Simulink SDK used, position estimation will not be discussed in details, but it is important to mention that there are several other approaches to estimate the position of the drone as well. The images captured using the bottom camera can be processed and used to approximate the horizontal speed of the drone, as used in AutoFlight ® Software, given that the ground over which the drone is flying has to have distinguishable features. Also, over-the-shelf low cost cameras or sensors can be used to estimate the position of the drone, as an example Kinect® can be used to capture the position of the drone in indoor applications as done by Yue [2].

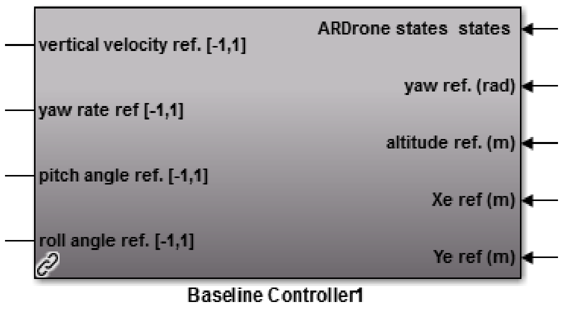

Figure 6.Baseline Controller Block

The controller block, compare it with the waypoints given by the user and uses the error to find the proper control inputs to the drone. The control inputs are not sent to the drone as x-y-z as this form is not understandable by the drone, rather, it is converted to roll, pitch, yaw rates and vertical speed.

Finally, the States Block is used to show the current height, x-y position and yaw angle of the drone on four scopes during execution of the model.

## IV. Vicon Capture System

The objective of this research is to get real measurements for the position of the drone in addition to the estimated position of the drone provided by the SDK. In order to achieve that, a position determination system should be used. It could be Global Positioning System (GPS), Inertial Navigation system (INS) or Motion Capture System. Since this project is performed indoor, a motion capture system was used. Motion capture (Mo-cap) or motion tracking is a process of capturing and recording the movements of people, objects or bodies using different types of techniques. It is used in many applications such as medical application, sport games, filmmaking, entertainment, etc. In this project, a Vicon motion capture system was used since it's available in one of the United Arab Emirates University Labs as shown in Figure 7. There are various techniques for the Vicon motion capture system and the one used in this project is the Optical-Passive Motion Capture technique. It's the mostly used technique due to its accuracy and flexibility. It uses high speed infrared cameras that track the motion of the reflective passive markers placed on the object.

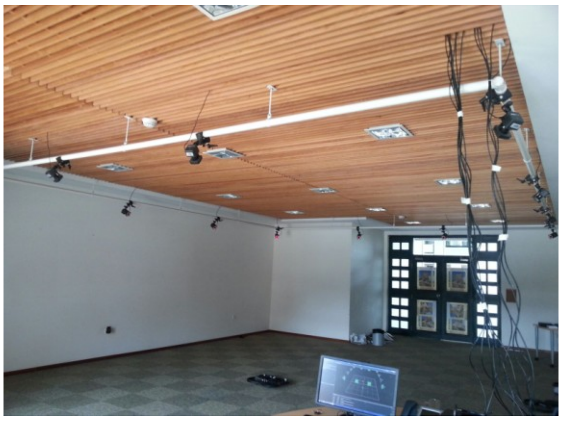

Figure 7. Vicon capture system lab in UAEU

The overall Vicon system consists of twelve Vicon Cameras, Motion Capture software, Calibration Wand, and Sync Box [6]. The Vicon cameras have a high speed and its sensors have a high resolution of 4MP and 16MP which increases the precision to obtain more details from the markers. There are various programs available to process the data depending on the application. In this project, Tracker software was used since it is the one recommended for engineering applications. The calibration wand is the tool used to calibrate the system's cameras. The sync box is the interface between the cameras and the base station (PC) which in turn will display the data obtained from the cameras.

## V. Results

First of all, the Vicon cameras were calibrated using the calibration wand and Tracker Software. Five reflective markers were placed in the body of the AR Drone, as shown in Figure 8, and it successfully tracked by Vicon cameras and added to the workspace. The workspace for the Tracker software is shown in Figure 9.

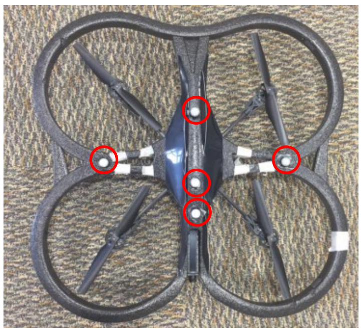

Figure 8. Parrot AR Drone 2.0 with four markers

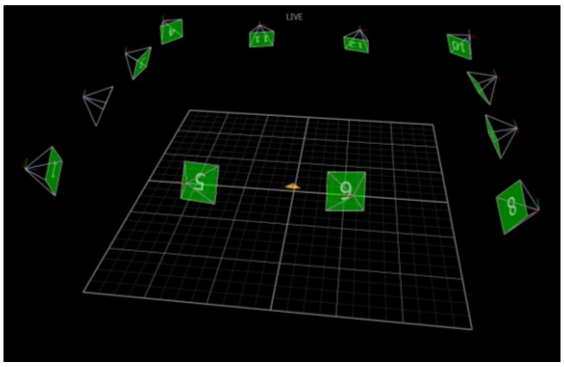

Figure 9. Tracker software workspace shows the Vicon Cameras and the AR.Drone at the origin.

In order to acquire the data from the AR Drone and the Vicon capture system, two communications should be established. The first one is between the AR Drone and the base station and the second one between the Vicon system and the base station. Because the AR Drone supports Wi-Fi Protocol and it has its own IP address (192.168.1.1), it will communicate with the base station wirelessly using Wi-Fi protocol through a router. The available AR Drone MATLAB / Simulink (SDK) stores the acquired data from the drone into workspace.

On the other hand, the Vicon Synx Box creates the interface between the Vicon cameras and base station with Tracker software. To acquire the measurements from the Vicon system, a MATLAB\Simulink block was built to create a communication. The MATLAB\Simulink uses Quanser Real-Time Control Software (QUARC) Target Library. It is a powerful tool that provides block sets such as hardware blocks and communication blocks. Our main target was to use the Vicon blocks and communication blocks provided by QUARC. Vicon blocks use DataStream API sand the Tracker software to get translation (x, y, z) and rotational (yaw, pitch, roll) measurements of the drone from the Vicon capture system. On the other hand, the communication blocks provide various communication protocols. Protocol parameters are specified with a Universal Resource Identifier (URI). In this project, Stream Client block used to connect the Vicon system (host) and receive the data from it using User Datagram Protocol (UDP). The Vicon and Stream Client blocks are shown below in Figure 10.

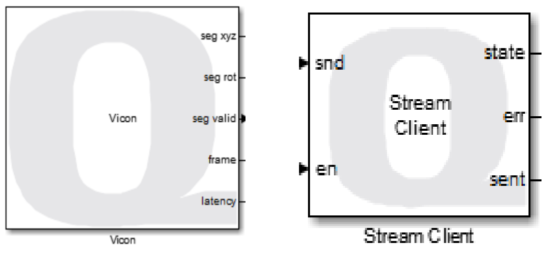

Figure 10. Vicon Block and Stream Client Block

After successfully running the Vicon Tracker program and getting the real position of the Parrot AR Drone 2.0 on the Vicon Simulink, the AR Drone Simulink needs to be linked to the Vicon Simulink. Packet Input block used to establish this connection and it uses UDP protocol.

In this way, the Vicon Simulink needs to be ran before running the AR Drone Simulink (SDK). The AR Drone Simulink (SDK) will receive six different data from the Vicon Simulink and by that the drone measurements is ready to be compared with the Vicon measurements.

Using the AR Drone Simulink control law, the drone has to track rectangular and circular trajectories. The code for the rectangular trajectory provided with the SDK, while the circular trajectory code developed as the following:

```
function [ waypoint ] = NewgetWaypointsCircle(
)
%GETWAYPOINTS Generates a list of waypoints for
the ARDrone
%   Each waypoint is a column vector that
contains the desired position of the
%   drone, desired heading angle, and waiting
time. The list of waypoints
%   is created combining the column vectors.

% This waypoints file will give the drone
waypoints of a circular path
% given its radius (input from the user). Total
number of points is nPoints.
% (nPoints/2) points for the upper half of the
circle and (nPoints/2) for the lower half

% Enter the radius:
r = 2;
% Enter the height
h = 1;
% Number of waypoints. Edit this value as
desired.
nPoints = 30; % HAS TO BE EVEN NUMBER!

waypointsListARDrone = zeros(5,nPoints);

% Edit the following entries for k
=1,2,...,nPoints
% waypointsListARDrone(:,k) = [ Xe (m), Ye (m),
h (m) ,yaw(rad),  waiting time (sec)
x = 0; % Coordinates of the first point
y = 0;

% Upper half of the circle
for k = 1 : 1 : nPoints / 2.0  % k is a counter
representing the number of the current waypoint
    waypointsListARDrone(:,k) = [x;y;h;0; 0] ;
    x = x + (4*r / nPoints); % x increments by
(2r / (nPoints/2))
    y = sqrt(r^2 - (x)^2);
end
 % Lower half of the circle
for k = nPoints / 2.0 : 1 : nPoints
    waypointsListARDrone(:,k) = [x;y;h;0; 0] ;
    x = x - (4*r / nPoints); % x decrements by
(2r / (nPoints/2))
    y = - sqrt(r^2 - (x)^2);
end

waypoint = waypointsListARDrone ;

end
```

The performance of the drone using SDK developed by Mathworks looks to be acceptable, yet the real measurements clearly show that the drone is not tracking its predefined path accurately. Running the two codes will allow the AR Drone to fly and track a square and a circular path as shown in Figures 12 and 13 respectively. The plots show that the AR Drone follows exactly the predefined waypoints. However, The Vicon cameras measurements were plotted as well for both paths to be compared to the results provided by the drone. The Vicon measurements show the accurate trajectory which the AR Drone actually tracked. Investigating has revealed several reasons for the inaccuracy of SDK, the main reason is the position estimation being inaccurate. Double integrating the acceleration and angle rates results in a notable bias error which makes the estimated position deviated from the real one by a considerable distance. Another reason which had a minor effect is the controller used in the baseline controller block. Proportional controller was used with a gain of unity or 0.5 in some cases which was not enough to have good controller and to keep the measured position as close as possible to the desired one. Another issue which has been faced during the implementation of this development kit was the inability to get the raw values of the sensors to have a chance to process theses values in a different way to get more accurate or realistic results. The way used to breakdown the received packet from AR Drone was complicated and not clearly documented which made it very difficult to modify it.

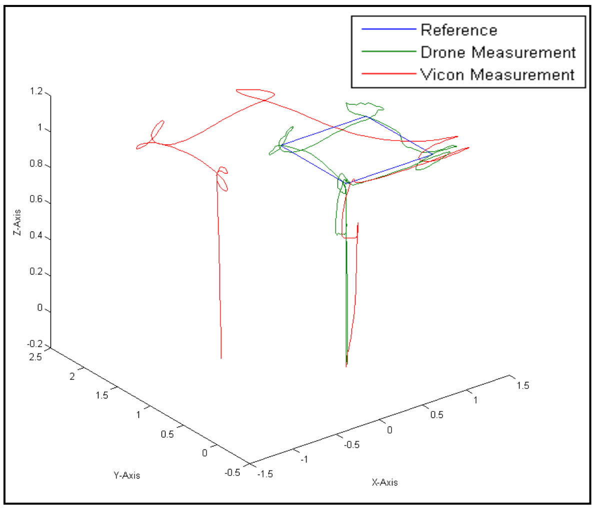

Figure 12. Square Trajectory measurements comparison

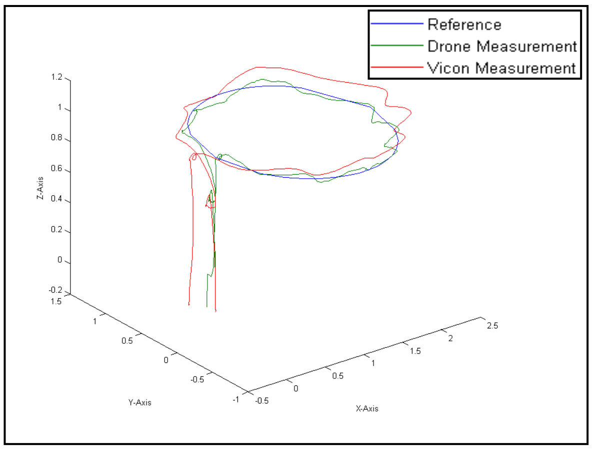

Figure 13. Circular Trajectory measurements comparison

## VI. Conclusion

To sum up, this project was done to control an indoor autonomous flight of Parrot AR Drone 2.0 using MATLAB/Simulink (SDK) and Vicon Camera System. The Vicon system was composed of twelve cameras that were calibrated and configured. The position of the Parrot AR Drone while hovering was estimated by the drone on-board sensors and transmitted wirelessly to the base station. Besides this, the real position of the parrot AR Drone was measured by the Vicon capture system and transferred to the computer/base station. The SDK and the Vicon MATLAB/Simulink were integrated together to make some comparisons between the Vicon cameras position measurement and the position estimation obtained by Parrot AR Drone on-board sensors. As a result, the Parrot AR Drone position estimation was found to be less accurate compared to the drone real position provided by Vicon cameras. As future perspective, the Vicon cameras measurements for the drone position will be integrated in the SDK's control algorithm to replace the inaccurate position estimation provided by the drone and improve it. Finally, this work will help in preparing a cooperative flight control of two or more quadrotors.

## *Acknowledgment*

This research is supported by the United Arab Emirates University, Al Ain, UAE, (Research Fund 31N214).

## *References*

[1] Jakob Julian Engel, "Jakob Julian Engel , Autonomous Camera-Based Navigation of a Quadrocopter , December 15, 2011," in *Autonomous Camera Based Navigation of a Quadcopter*, 2011.

[2] Sun Yue, Project, Illinois, Urbana-Champaign, 2012.

[3] Michael David Schmidt, "2011 SIMULATION AND CONTROL OF A QUADROTOR UNMANNED AERIAL VEHICLE," University of Kentucky, 2011.

[4] "AR Drone Simulink Development-Kit V1.1 - File Exchange - MATLAB Central." [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/43719-ar-drone-simulink-development-kit-v1-1. [Accessed: 16-Sep-2015].

[5] Lukas Lao Beyer, "Connecting to the AR Drone," *http://autoflight.readthedocs.org/en/latest/gettingstarted.html*. .

[6] "What is motion capture | VICON." [Online]. Available: http://www.vicon.com/what-is-motion-capture. [Accessed: 16-Sep-2015].