# Quadcopter control using Android based sensing

AUGUST BJÄLEMARK
Lund University
Department of Automatic Control
Ole Römers väg 1, SE 223 63, Lund
SWEDEN
agge.bjalemark@gmail.com

HANNES BERGKVIST
Lund University
Department of Automatic Control
Ole Römers väg 1, SE 223 63, Lund
SWEDEN
hannes.bergkvist@gmail.com

*Abstract:* This paper investigates the concept of a quadcopter implemented using the sensors and computational power of a smartphone. The main goal is to give an example of the possible applications of external hardware combined with commercial off the shelf (COTS) electronics. Practical problems that required attention were implementation of sensor fusion using the smartphone's sensors and the controller as a smartphone app. Attention was also paid to the real-time aspects of the Android operating system.

*Key–Words:* Android, Quadcopter, COTS, real-time

## 1 Introduction

External hardware combined with commercial off the shelf (COTS) electronics is a concept that enables development of simple and low cost devices that obtain functionality when connected to the COTS-product. A smartphone is an example of COTS electronics with multiple features such as:

- Computational power
- Sensors
- Wireless connectivity
- Touch screen
- Camera
- Easy development and distribution of third party applications

As the number of smartphones increases [1], it is of interest to investigate the concept of combining a smartphone with external hardware. A quadcopter is a good platform for this investigation for the following reasons:

- It needs sensors (accelerometer, magnetometer and
gyroscope)
- It needs computational power (Processor)
- It needs algorithms for control and functionality (Easy development of applications)
- It benefits from wireless connectivity (Wi-Fi, Bluetooth, 3G/4G)
- It benefits from other sensors available (pressure sensor, GPS, camera)

The quadcopter was chosen as it has hard real-time requirements [2], and the ambition is to give a benchmark for the capabilities of this concept.

Since Android is the most used mobile operating system [1], an Android based smartphone is suitable for this investigation. Android is an operating system based on Linux, made primary for touch screen mobile devices [3]. In 2013 there were more than 900 000 000 devices activated with Android as operating system [4].

A quadcopter is an aerial vehicle with four rotors. The design of a quadcopter enables it to fly without any other moving parts except the rotors. The speed of the four rotors determines all movements of the quadcopter. It requires sensors for angular estimation and a control system to stabilize it during flight. This is usually implemented with a dedicated microprocessor for signal processing and control algorithms. Accelerometer and gyroscope sensors connected to the microprocessor are then used to estimate the state of the quadcopter [5].

In this project the control and signal processing algorithms are implemented with an application that is installed on the smartphone. The sensordata is acquired from the accelerometer, gyroscope and magnetometer featured in the phone.

If a smartphone is available, this approach enables design of a very simple and low cost quadcopter. Additionally it becomes easy to develop, distribute and update the software. If further functionality is desired it is easy to make use of other features on the smartphone, such as camera and image processing capabilities.

Android based devices are usually not considered to be adequate for hard real-time applications [6]. One major issue is the consistency of Android smartphone applications. Problems with consistency arise from different areas, including:

- Difficulty to take into consideration all the details about how the system is implemented, for example services that require large amounts of processing power that[7].

- Andriod smartphone applications are based on Java. One specific issue is Java's garbage collector. Its task is to locate allocated memory that is not going to be used again, and then deallocate it. This process uses up resources and it is hard to predict when it occurs.

Figure 1 shows how different unexpected behaviors explained above affect the time it takes to complete a control cycle.

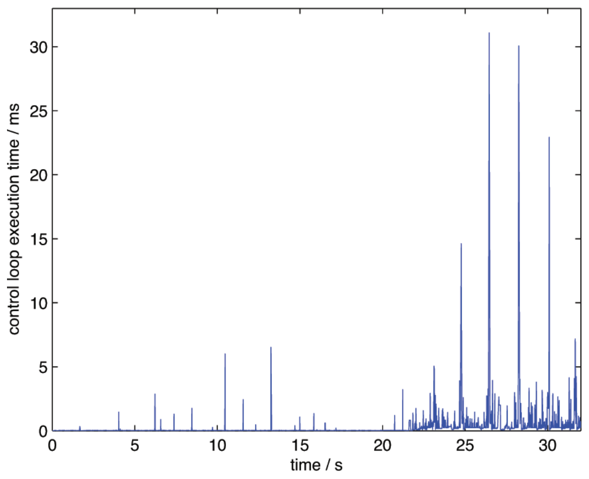

**Figure 1:** The time it takes to perform all calculations during a control cycle at different moments in time. After 22 seconds some additional code is running in the control loop, with 10 objects being allocated every cycle. As seen in the graph the time to complete a cycle is now more often 1ms or above then before the 22 seconds mark. There are also several big spikes, these could be a result of the garbage collector.

Android has support for C/C++ development using the Native Development Kit (NDK), which provides an alternative for faster algorithm execution time [8]. This involves writing C/C++ functions that are later going to be called from the Java program. Using native calls (calls to C/C++ functions) will however affect the performance [9]. It is therefore not always better to use native functions over regular Java functions. Example of good candidates for NDK are CPU-intensive operations that don't allocate ma lot of memory [10]. Since this project does not require any particular CPU-intensive algorithms, there is no benefit in utilizing NDK and it was decided to use standard Java only.

In 2013 there were several commercial and research products based on smartphones. One of them is Romo [11], a commercial robot based on an Iphone. Romo can drive around by itself, communicate with another Iphone and perform minor image processing. An example of a research application is PhoneSat [12], NASA's satellites based on Android phones. The PhoneSat project strives to decrease the cost of satellites while not sacrificing performance.

## 2 Experimental setup

The smartphone used in this project is the Samsung Galaxy S3, which was released in May 2012 [14]. The quadcopter is custom built around a light weight frame. It has four brushless DC motors with 250 mm propellers as actuators, see Figure 2.

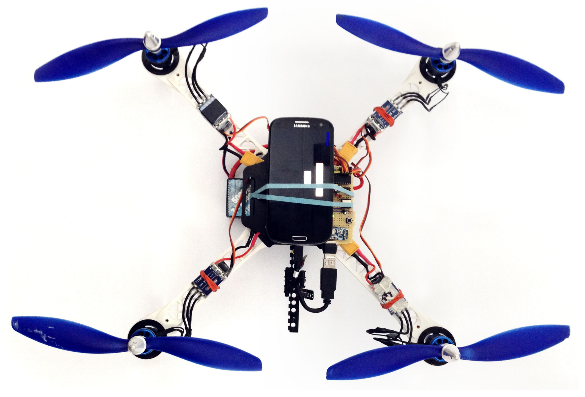

**Figure 2:** The quadcopter built in this project. The smartphone used for IMU-measurements, filtering and control algorithms can be seen on top of the frame.

The thrust of a motor is determined by a pulse width modulated (PWM) signal to a motor controller, which controls the three phase current to the motor. The PWM signal is generated by an Atmega88 microprocessor which in turn communicates with the smartphone using a USB to serial converter. In order to signal the smartphone to act as master, a special USB cable was used in the communication. This type of cable is called USB on-the-go (or USB-OTG). The architecture of the cable is similar to a regular USB cable, with the only difference being that on the smartphone end, the sense connector is connected to ground, see Figure 3. The motor controllers and the Atmega88 circuitry are the only external electronics used in the system. The experiments in this work are performed with the quadcopter flying, or at surface level mounted on a balljoint. Power is suplied by a 7.2V Lithium polymere (LiPo) battery during flight. For experiments performed at surface level a power supply unit (PSU) is used to power the system. When using the balljoint the number of degrees of freedom decreases from six to three. This enables more controlled and safe testing of new algorithms before a flight experiment.

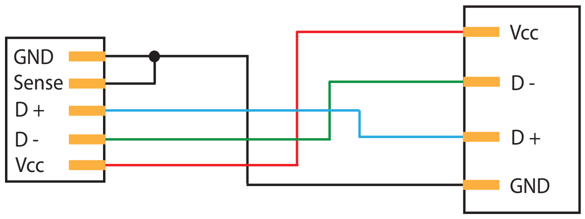

**Figure 3:** USB OTG cable. The sense connector is connected to ground to signal the smartphone that it should act as master.

The smartphone communicates with a PC over a wireless network. This connection is used to send data from the smartphone to the PC, for example live images or logdata. It is also used to send commands from the PC to the quadcopter, for example angle references. The communication from the PC to the smartphone uses the TCP protocol [13], since it is important that all the commands arrive. The same TCP connection is used to measure the latency. The video streaming on the other hand is implemented using the UDP protocol [13], since it is more important that the new frames arrive instead of the old one being resent.

## 3 IMPLEMENTATION

The implementation of a controller for a quadcopter using Android based sensing requires attention to the following areas.

### 3.1 Modeling

The model is based on the general model of the dynamics of a quadcopter presented in [15]. It is modified for a quadcopter in X-configuration, see Figure 4.

The desired quantities of the system to control are the angles around each axis. The input signals are the torque values around each axis, see (1). In order to compensate for loss of lifting force when the quadcopter is tilted, attention is also paid to the dynamics along the Z-axis, see (1d). The relationship between the torque values and the thrust values of the motors

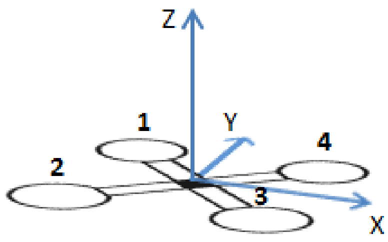

**Figure 4:** The coordinate system of the quadcopter used in this project. The forward movement is in the Y direction (X-configuration).

are presented in (2).

$$\ddot{\theta} = \frac{I_{yy} - I_{zz}}{I_{xx}}\dot{\phi}\dot{\psi} - \frac{J_{TP}}{I_{xx}}\dot{\phi}\Omega + \frac{U_2}{I_{xx}} \tag{1a}$$

$$\ddot{\phi} = \frac{I_{zz} - I_{xx}}{I_{yy}}\dot{\theta}\dot{\psi} - \frac{J_{TP}}{I_{yy}}\dot{\theta}\Omega + \frac{U_3}{I_{yy}} \tag{1b}$$

$$\ddot{\psi} = \frac{I_{xx} - I_{yy}}{I_{zz}}\dot{\theta}\dot{\phi} + \frac{U_4}{I_{zz}} \tag{1c}$$

$$\ddot{Z} = -g + \cos\theta\cos\phi\frac{U_1}{m} \tag{1d}$$

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{1e}$$

Where $\omega$ is the angular velocity of each propeller, $\theta$, $\phi$, $\psi$ the angle around the X-, Y- and Z-axis respectively, $J_{TP}$ the total rotational moment of inertia around the propeller axis and $U_{1-4}$ are the inputs to the system described in (2).

$$\begin{aligned} U_1 &= T_1 + T_2 + T_3 + T_4 \\ U_2 &= l(T_4 + T_1 - T_2 - T_3) \\ U_3 &= l(T_1 + T_2 - T_3 - T_4) \\ U_4 &= R_2 + R_4 - R_1 - R_3 \end{aligned} \tag{2}$$

Where $l$ is the distance between the center of a propeller to the center of mass of the quadcopter, $T_{1-4}$ the thrust from the different motors and $R_{1-4}$ the torque they exert around the Z-axis. The controller will be derived under the assumption that the quadcopter will mostly hover with all its angles and angular velocities equal to zero. The model can thus be simplified to (3).

$$\begin{aligned} &\ddot{\theta} = \frac{U_2}{I_{xx}}, \ddot{\phi} = \frac{U_3}{I_{yy}}, \ddot{\psi} = \frac{U_4}{I_{zz}} \\ &\ddot{Z} = -g + \cos\theta\cos\phi\frac{U_1}{m} \end{aligned} \tag{3}$$

## 3.2 Control

The models for the angles are of second order, therefore the PID controller structure was chosen. A D-part is usually implemented with differentiation of the control error. This differentiation amplifies high frequency noise and a low-pass filter is required. However, in this case it is possible to acquire the angular velocity directly from the gyroscope, therefore no filter is used. To avoid differentiation of step references, the D-part only acts on the angular velocity and not on the derivative of the error. An I part is used to remove stationary errors. The controller is shown in (4).

$$\begin{gathered} U(s) = (R(s) - Y(s))K_P + \\ (R(s) - Y(s))\frac{K_I}{s} - sY(s)K_D \end{gathered} \tag{4}$$

Here $K_P$, $K_I$ and $K_D$ are the parameters for the P, I and D part respectively. To handle actuator saturation, anti-windup is used.

The control loop mean cycle time is limited to 10 ms due to the sensor fusion cycle time being 10 ms. Calculation of control signals:

```
u = K * (refAngleY - getAngle())
    -getAngleVel() * D
    +Iangle * I);
```

Integration and saturation:

```
Iangle += refAngle - getAngle();
if (Iangle*I > satValue)
    Iangle = satValue/I;

if (Iangle*I < -satValue)
    Iangle = -satValue/I;
```

## 3.3 Sensor fusion

The system makes use of the accelerometer, magnetometer and gyroscope data to estimate the rotation around the X-, Y- and Z-axis, see Figure 5.

The accelerometer measures the gravitational acceleration as reference to calculate an estimate of the angle around the X-, and Y- axis, as shown in (5).

$$\begin{aligned} Y_{\text{angle}} &= \arctan(X/Z) \\ X_{\text{angle}} &= \arctan(Z/Y) \end{aligned} \tag{5}$$

Here, X, Y and Z are the accelerometer value in the respective directions.

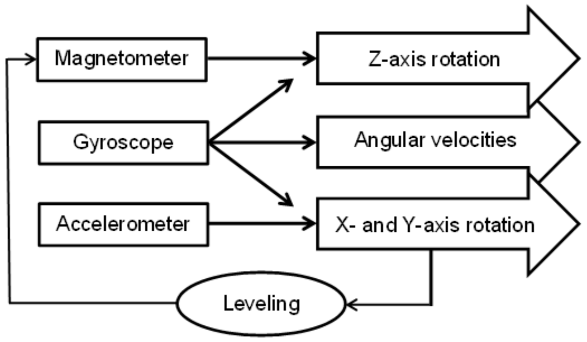

**Figure 5:** Usage of sensors.

However, during flight the quadcopters acceleration will affect the measurement and make it less useful [16].

The magnetometer makes use of the earth's magnetic field as a reference to calculate an estimate of the angle around the Z-axis. Since the magnetic field will vary with rotation around Y- and X-axis, the sensor is leveled using the offset obtained from accelerometer and gyroscope, see (6), (7) and (8).

$$\begin{bmatrix} B'_x \\ B'_y \\ B'_z \end{bmatrix} = \begin{bmatrix} cy & 0 & sy \\ 0 & 1 & 0 \\ -sy & 0 & cy \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & cx & -sx \\ 0 & sx & cx \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} B'_x \\ B'_y \\ B'_z \end{bmatrix} = \begin{bmatrix} cy & 0 & sy \\ cx * sy & cx & -sx * cy \\ -cx * sy & sx & cx * cy \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} \tag{7}$$

$$Z_{\text{angle}} = \text{atan2}(B'_y, B'_x) \tag{8}$$

Where cx, cy, sx, sy are cos(x), cos(y), sin(x) and sin(y) respectively and atan2 is the four quadrant arctan function

A gyroscope measures the angular velocity in reference to its own coordinate system instead of an external factor as with the accelerometer (gravity) and the magnetometer (magnetic fields). To obtain the orientation the signal is integrated. To get accurate orientation of the device the sensor data is filtered and combined with regard to its strengths and weaknesses. This sensor fusion consists of two different complementary filters, one for the accelerometer/gyroscope fusion and one for the gyroscope/magnetometer fusion.

The accelerometer data has high frequency noise, which can be attenuated with a low pass filter. This makes the estimate more accurate, but adds a weakness in making it less responsive to fast changes. The gyroscope on the other hand is fast and accurate but when integrated to estimate the angle, it has a drift error. This can be removed with a high pass filter. The magnetometer has, similar to the accelerometer high frequency noise, which can be attenuated with a low pass filter.

When using accelerometer and gyroscope measurements, their respective filtered signals complement each other and produces both fast and long-term accurate estimate of the rotation around the X- and Y -axis. The magnetometer is combined with the gyroscope in the same way to calculate an estimate of the rotation around the Z-axis. The complementary filter used is presented in (9).

$$\theta(z) = \frac{1-\alpha}{1-\alpha z^{-1}} A(z) + \frac{\alpha}{1-\alpha z^{-1}} hG(z) \quad (9)$$

Here, A(z) is the angle estimation from the accelerometer, G(z) is the angular velocity acquired from the gyroscope, $h$ the sampling period and $\alpha$ the filter parameter. The sensors of the Samsung Galaxy S3 have a mean sampling frequency of 100 Hz. The mean cycle time of the sensor fusion is 10 ms.

The accelerometer/gyroscope fusion filter:

```
fAngle = alpha
       * (fAngle + getGyroVal()
       * (getTimestamp() - tFuse))
       + (1-alpha)*getAccAngle();
tFuse=getTimestamp();
```

## 4 RESULTS

The external measurments were taken using a Nikon K600 camera registering the positing of throbbing LEDs fastened on the quadcopter. Measurments were taken with a frequency of 100 Hz.

To evaluate the performace of the complementary filter, an experiment was performed, see Figure 6. In this experiment the quadcopter was mounted on the balljoint. The quadcopter was first in an approxematly horizontal position, after 10 seconds it was manually tilted an angle of 0.4 radians around the Y-axis. After about 20 additional seconds it was tilted various angles and then returned to its original position. Angular estimates were collected from the complementary filter and compared with external measurements from the Nikon system. To test the quadcopter's behaviour during flight, one more experiment was performed, see Figure 7. In this experiment the quadcopter is hovering with a reference angle equal to zero. After about five seconds a short angle reference was given. The angle was estimated using the complementary filter. The filter parameters were the same as in the previous experiment. $K_P$, $K_I$ and $K_D$ was now set to 100, 0 and 50 respectively. The result shows that the error is not converging, but remains bounded. This is not the case when the quadcopter is mounted on the balljoint. The major contributing factor to the nonconverging error are vibrations in the frame. The angle is fluctuating around zero while hovering, therefore no integrator in the controller was required.

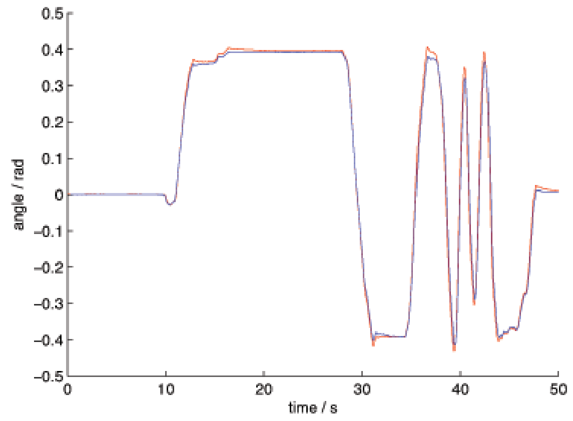

**Figure 6:** Comparison between the estimated angle from the complementary filter (red) and the measured angle from the Nikon system (blue).

$\alpha$ and $h$ in the complementary filter was set to 0.006 and 0.01 s respectively. The maximum occurring error is 0.025 radians.

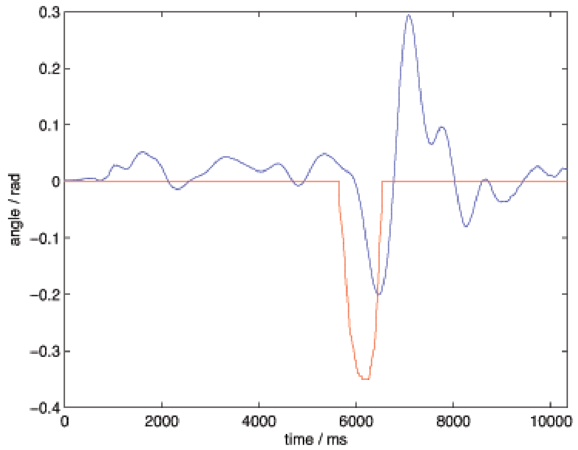

**Figure 7:** Angular reference (red) and measured angle using the complementary filter (blue).

## 5 CONCLUSION AND FUTURE WORK

### 5.1 Conclusion

It is possible to implement a quadcopter using the sensors and computational power of a smartphone. The smartphone used was a Samsung Galaxy S3. The application was implemented in Java, without the use of NDK. The quadcopter utilized the accelerometer, gyroscope and magnetometer of the smartphone to estimate the orientation. To improve the estimate sensor data was fused and filtered with two complementary filters. The sensors proved to be accurate and fast enough for the application. It was sufficient to use PD controllers to stabilize the quadcopter.

### 5.2 Future work

Suggestions of possible improvements and features that could be implemented:

- Image processing. The camera in the smartphone used in this project can take about 16 frames per second at a resolution of 640x480.
- Steering and streaming of images over a 3G network.
- Better hardware. The frame vibrates because of the motors, which disturbs the sensor readings.
- Compensations for the magnetic field generated by the motors. Every motor creates a magnetic field that is dependent on its angular velocity. This phenomenon disturbs the magnetometer estimations of the angles. However, as the relationship is deterministic, it should be possible to compensate for it.

## 6 Acknowledgement

Thanks to Jerker Nordh and Anders Robertsson for good supervision and availability when help was needed.

*References:*

[1] Gartner, Newsroom, 2013. http://www.gartner.com/newsroom/id/2623415 (visited 2013-08-05)

[2] J.J. Corona-Sánchez. H. Rodríguez-Cortés. *Experimental real-time validation of an attitude nonlinear controller for the quadrotor vehicle*, IEEE International Conference on Unmanned Aircraft Systems (ICUAS), pages 453 - 460, 2013.

[3] Open handset alliance, Android overview, http://www.openhandsetalliance.com/android_overview.html (visited 2013-08-10)

[4] Android, http://www.android.com/ (visited 2013-08-21)

[5] E. Davis, B.E. Nizette and Changbin Yu *Development of a low cost quadrotor platform for swarm experiments*, IEEE International Chinese Control Conference (CCC), pages 7072 - 7077, 2013

[6] L. Perneel, H. Fayyad-Kazan and M. Timmerman *Can Android be used for Real-Time purposes?*, IEEE International Conference on Computer Systems and Industrial Informatics (ICCSII), pages 1-6, 2012

[7] Android, Developer resources, http://developer.android.com/guide/components /services.html (visited 2013-08-21)

[8] Ki-Cheol Son and Jong-Yeol Lee *The Method of Android Application Speed up by using NDK*, IEEE International Conference on Computer Systems and Industrial Informatics (ICCSII), pages 382-385, 2011

[9] Android, Developer resources, http://developer.android.com/training/articles/ perf-tips.html (visited 2013-08-21)

[10] Android, Developers https://developer.android.com/tools/sdk/ndk/in dex.html (visited 2013-08-21)

[11] Romo, meet romo. http://romotive.com/meet-romo (visited 2013-08-15)

[12] Phonesat, 2013. http://www.phonesat.org/ (visited 2013-08-15)

[13] T.Gopinath, A.S.Rathan Kumar, Rinki Sharma *Performance Evaluation of TCP and UDP over Wireless Ad-hoc Networks with Varying Traffic Loads*, IEEE International Conference on Communication Systems and Network Technologies (CSNT), pages 281-285, 2013.

[14] Samsung, Galaxy S3, 2012.
http://www.samsung.com/global/galaxys3/
specifications.html
(visited 2013-08-21)

[15] T. Bresciani *Modelling, Identification and control of a quadrotor Helicopter*, Master Thesis, Lund Institute of Technology, 2008.

[16] P. Martin and E. Salaun *The True Role of Accelerometer Feedback in Quadrotor Control*, IEEE International Conference on Robotics and Automation (ICRA), pages 16231629, 2010.