The Potential for UAV Technology to Assist in Sheep Management in the Scottish Highlands

Marc-Alexandre Favier[1], Richard C.P. Green[1], Andreas Linz[2]

[1] *Harper Adams University, Edgmond, Newport, Shropshire, TF10 8NB, UK*
Email: favier.marcalexandre@gmail.com
[2] *Hochschule Osnabrück, Fakultät IuI, Albrechtstraße 30, 49076 Osnabrück*

Summary: In the Scottish Highlands, the reform of the Common Agricultural Policy has been encouraging farmers to reduce or to stop their sheep farming. The remaining sheep tend to roam further into the less grazed areas. This is possible because the fences in the hill regions are often not fully stock-proof. This generates more work when gathering the sheep. In October 2012 Harper Adams University started investigating the use of Unmanned Aerial Vehicles (UAVs) to locate and round up sheep.

A functional prototype system was developed using a low cost, off-the-shelf, four propeller UAV, the AR.Drone2.0 made by Parrot, controlled from a laptop base station running LabVIEW and Robotics Operating System (ROS) connected to the UAV via Wi-Fi.

Sensors available on the drone included an altimeter, an ultrasonic sensor used to measure height, a three axis electronic gyrocompass, a three axis accelerometer and a magnetometer. Also fitted to the AR.Drone2.0 were two cameras, one looking forward and one vertically downwards. A Windows7 laptop containing a Linux/Ubuntu virtual machine was used as the base station. National Instruments' (NI) Vision Assistant Software, NI's LabVIEW and Clearpath Robotics' ROS toolkit were installed on the laptop in Windows whilst ROS was installed in the Linux/Ubuntu virtual machine. Clearpath Robotics' ROS toolkit was used to interface LabVIEW to ROS. LabVIEW was used for image analysis and position calculations using a closed-loop feedback Proportional Integral (PI) control system. Navigational information was passed to ROS which communicated with drone's internal processor.

An image processing algorithm has been developed with the assistance of NI's Vision Assistant Software to detect uniquely shaped targets. Initial testing, conducted indoors in a sports hall, using a red circular target, has shown that the drone is able to use this algorithm to locate and track a target. Further tests with a static camera have been conducted to detect sheep grazing in a field. To increase the contrast between the animals and the grass, the image processing algorithm contains a routine which enhances the green colour of the grass. It is planned to conduct further tests using the UAV's embedded cameras to locate and track sheep outdoors in fields.

Keywords: Image processing, Unmanned Aerial Vehicle, UAV, Sheep location.

Zusammenfassung: *Im schottischen Hochland hat die gemeinsame Agrarpolitik dazu geführt, dass viele Landwirte die Schafhaltung reduziert oder sogar aufgegeben haben. Allerdings tendieren Schafe dazu, dann in die weniger beweideten Nachbarregionen zu ziehen. Dies ist möglich, da die Zäune in den Bergregionen oft nicht voll funktionsfähig sind. Dies resultiert dann in einem erhöhten Arbeitsaufwand, wenn die Schafe zusammengetrieben werden. Im Oktober 2012 startete die Universität Harper Adams eine Untersuchung über die Nutzung von unbemannten fliegenden Systemen für die Lokalisierung und das Zusammentreiben von Schafen.*

Ein funktionaler Prototyp wurde entwickelt und besteht aus einem kostengünstigen Quadrokopter, dem AR.Drone2.0 von der Firma Parrot. Die Drohne wird über eine Basisstation gesteuert und die Robotic Operating System (ROS) Platform ist mit der Drohne über Wi-Fi verbunden.

Die in der Drohne verfügbaren Sensoren sind: Altimeter, Ultraschall Sensor zur Messung der Höhe, ein drei-achsiges Gyroskop und ein drei-achsiger Magnetometer. Die beiden Kameras sind fixiert, eine blickt nach vorne, eine andere nach unten.

Ein mit Windows7 betriebener Laptop und eine unter Linux-Ubuntu durchgeführte virtuelle Maschine wurden als Basisstation verwendet. Während LabVIEW (National Instrument: NI), Vision Assistant Software (NI) und das ROS Toolkit von der Firma Clearpath Robotics auf dem Laptop installiert wurden, wurde ROS auf der Linux Ubuntu virtuellen Maschine installiert. Das Toolkit der Firma Clearpath Robotics gewährleistete die Verbindung zwischen LabVIEW und ROS. LabVIEW wurde für die Bildverarbeitung und für die Lageregelung benutzt. Die Navigationsdaten aus LabVIEW wurden an die ROS Platform übertragen und anschließend zu dem Prozessor der Drohne.

Mithilfe der NI Vision Assistant Software wurde der Bildverarbeitungsalgorithmus zur Positionierung einzelner Zielobjekten entwickelt. Die ersten Tests, die in einer Sporthalle mit einem roten kreisförmigen Zielobjekt durchgeführt wurden, haben gezeigt, dass das UAV System in der Lage ist, ein Zielobjekt zu lokalisieren und diesem Objekt zu folgen. Mehrere Tests wurden durchgeführt, damit das grasende Schaf detektiert werden kann. Um den Kontrast zwischen dem Tier und dem Gras zu verstärken, besteht der Algorithmus aus einer Routine, welche die grüne Farbe des Grases intensiviert. Es ist geplant zusätzliche Versuche zur Lokalisierung und zum Tracking von Schafen mit der Drohne draußen in dem Feld durchzuführen.

Deskriptoren: *Bildverarbeitung, UAV, Regelungstechnik, Schaffen Haltung*

1 Introduction

In the Scottish Highlands, the number of sheep has been dramatically reduced over the last 10 years (THOMSON *et al.* 2011). In fact since the reform of the Common Agricultural Policy in 2003, farms have not been funded anymore per number of animals but the size of the land. This measure encouraged farmers to reduce or stop sheep farming and to use the land differently in order to keep their single farm payment (WATERHOUSE *et al*. 2007). As the neighbouring land of the remaining sheep farmers is often not used for livestock anymore, the remaining flocks tend to roam into the neighbouring land. This generates more work when gathering the sheep. Since October 2012, Harper Adams University has been investigating ways to support shepherds and farm managers. The objective of this project was to produce a functional prototype UAV system for locating and tracking lost sheep. The paper provides an overview of the process involved in the research from the architecture design to testing.

2 Functional prototype

2.1 Main Architecture

The architecture for the prototyping phase of the product is as follows. The functional prototype comprises of a low cost, off-the-shelf, four propellers UAV, an AR.Drone2.0 made by Parrot and a laptop base station on which ROS (which stands for Robot Operating System) and LabVIEW have been installed.

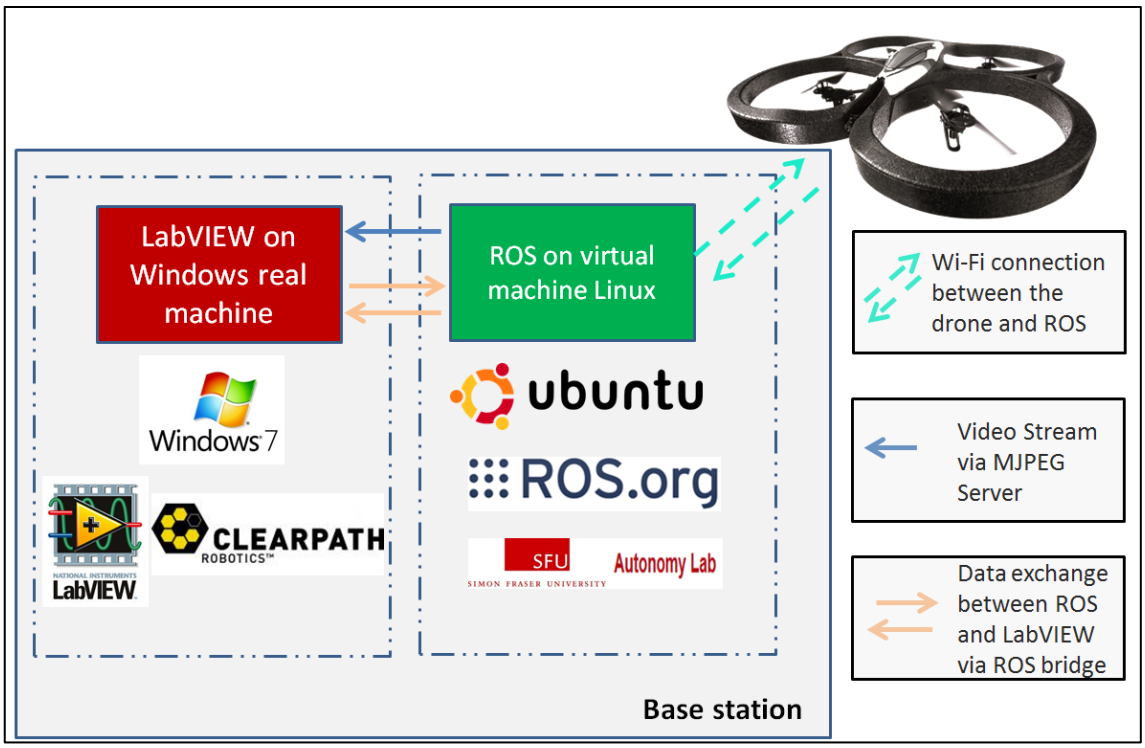

Figure 1: Functional prototype architecture

As ROS and LabVIEW work with two different operating systems, the former was installed in a Linux-Ubuntu virtual machine and the latter in the Windows7 machine. An interface was created finally between the two. This architecture is outlined in **Figure 1**.

2.2 AR.Drone2.0

The AR.Drone2.0 was chosen as the functional prototype for a number of reasons. Firstly, it is a low-cost, off-the-shelf drone and its popularity for two years also makes web research easy. Even if the end product will use a more robust drone, the AR.Drone2.0 allowed the authors to test the architecture without risking an expensive aerial vehicle. There are also technical reasons for choosing the AR.Drone2.0. It is equipped with multiple sensors whose main function is to determine the orientation in space of the drone: pitch, roll and yaw. The height of the drone is determined using an ultrasonic distance sensor and a pressure sensor. Attitude and velocity are determined using an inertial measurement unit (Accelerometer and Gyroscope) and a camera to measure ground speed. Data was transmitted to the base station via Wi-Fi.

2.3 Base station: ROS on Ubuntu virtual machine

The ardrone-autonomy ROS driver (MONAJJEMI 2012) developed by the Autonomy Lab at Simon Fraser University, Canada, was preferred to the SDK and project example provided by Parrot, the manufacturer of the AR.Drone. There were several reasons for this. Firstly, ROS is an open source Robot Platform which is rapidly becoming the preferred software development tool for robotics. Secondly, it is far easier to link to other image processing (openCV) or SLAM software using ROS (LINZ & RUCKELSHAUEN 2012, QUIGLEY *et al.* 2009). Finally, it was possible to connect ROS to LabVIEW via the Clearpath Robotic ROS Toolkit (CLEARPATHROBOTICS 2012). Therefore, ROS and the ardrone-autonomy driver were installed on the Linux-Ubuntu virtual machine in the laptop base station.

Before the Linux-Ubuntu virtual machine can be used *Roscore* and *Rosdriver* commands need to be executed to enable ROS to control the drone. Once that has been done it is possible to send basic commands like *take-off* or *land* to the drone. The connection between the virtual- and the real-machine is made using the *Rosbridge* and *MJPEG server* commands thereby allowing respectively the transmission of navigation data and image data(CLEARPATHROBOTICS 2012, PITZER 2011).

2.4 Base station: LabVIEW on Windows7 machine

Whilst developing the program in LabVIEW on the Windows7 machine, great attention was given to achieving a scalable, maintainable and efficient program. A good project structure, the use of state machines and simultaneously running VIs (Virtual Instrument, which is a LabVIEW file containing the code) are amongst the most important practices advised when programming in LabVIEW (BLUME 2007). A LabVIEW project with the file extension .lvproj which contains four different directories was created. The contents of these directories; “start”, “control”, “vision” and “navigation data” are outlined below.

The “start” directory contains all of the files required to start the functional prototype. The “control” directory comprises of the state machines which allows the user to switch between the different flying modes: Gamepad or Tracking. It is also possible to execute simple command such as take-off and landing separately. In the “vision” directory are the files required to acquire and process the video signal from the Virtual USB Camera. Finally in the “navigation data” directory there is only one file, which extracts the navigation data from the virtual machine. An example of the use of the ROS toolkit in a LabVIEW program file is shown in **Figure 2** below. This shows ‘takeoff.vi’ which is used to make the drone take-off.

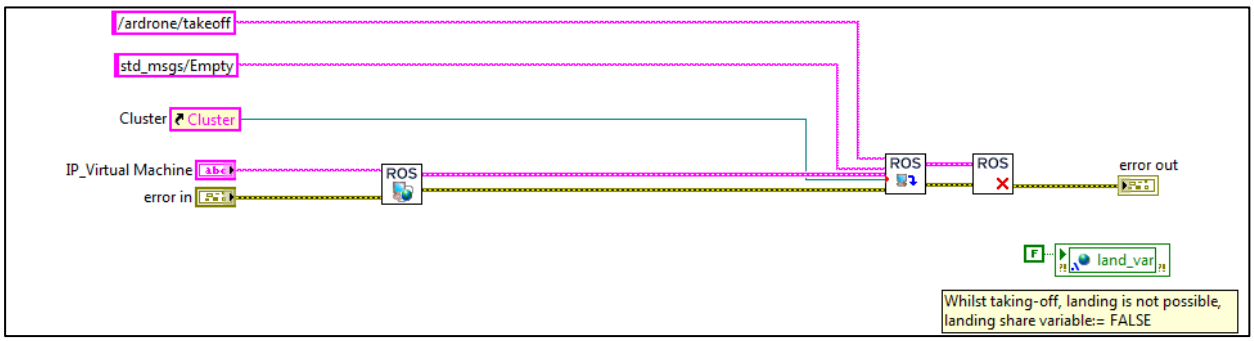

Figure 2: takeoff.vi, file part from the “control” directory, using the Clearpath robotics ROS-Toolkit

3 Machine Vision

3.1 Image Acquisition

Although the ROS Toolkit makes the acquisition of ROS messages from the virtual machine possible, it does not take charge of the video acquisition from ROS. This was solved by using ROS MJPEG server. This function, which is executed in the virtual machine, sends the video stream from the drone to an MJPEG server, accessible at the following address: http://192.168.1.3:8080/stream?topic=/ardrone/image_raw. A way of accessing this live stream in LabVIEW was found on the LabVIEW developer website. This involved

using IP camera adapter software to stream the image data to a virtual USB camera which LabVIEW recognises (**Figure 3**).

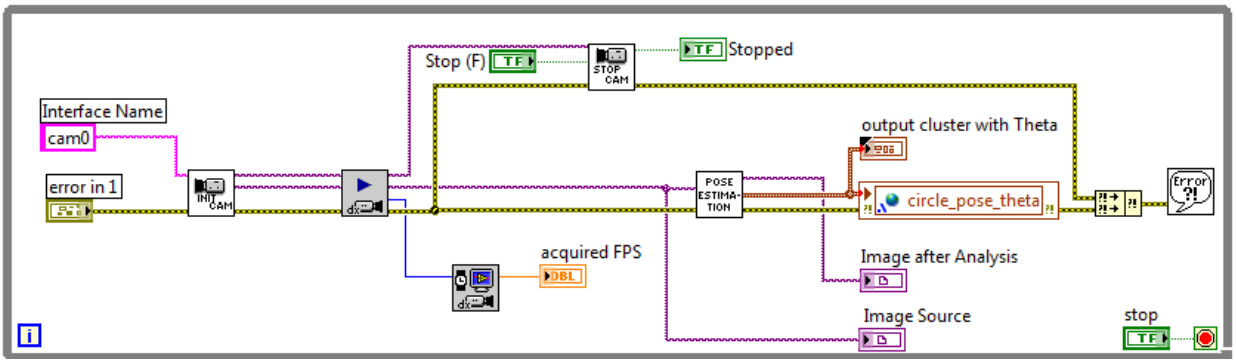

Figure 3: Diagram of vision.vi, part from the “vision” directory

3.2 Image Processing: Simple target detection

As previously mentioned above, the objective of this study is to use a UAV to locate and track sheep. Tracking a red target was the first step in addressing this problem. Firstly it enabled the UAV system to be tested using a simple target. Secondly, if the sheep recognition algorithm does not work satisfactorily, painting a red disk on the back of a sheep would allow other parts of the control software to be tested. Use of the Vision Assistant Module from National Instrument enabled the rapid development of an image processing algorithm capable of recognising a circular object. This is a two stage process (**Figure 4**). First the luminance plane is extracted from the original image. Secondly a routine detecting circular edges determines the position in the image of the red disk and its radius in pixels (NATIONAL INSTRUMENT 2012). Finally the Vision Assistant script was converted into a LabVIEW program.

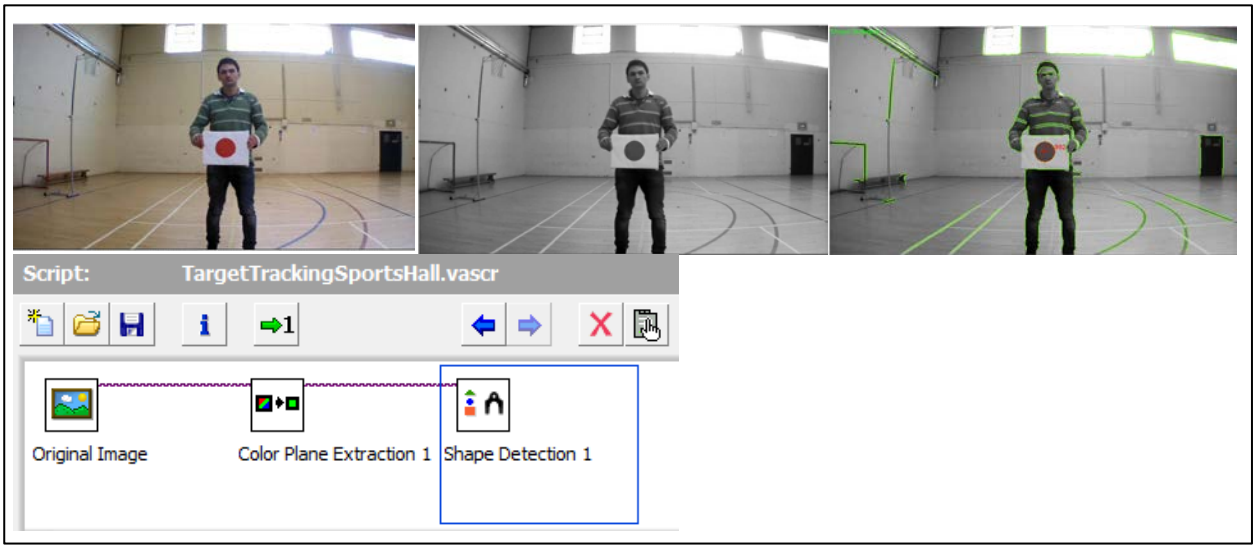

Figure 4: Target Recognition Development with NI Vision Assistant

3.3 Visual servo control

Once the image from the drone was acquired in LabVIEW, a new LabVIEW file known as a VI (see above 2.4) was developed for the target tracking. Two different methods were investigated to perform tracking: Position-Based Visual Servo (PBVS) and Imaged-Based Visual Servo (IBVS) (CORKE 2013). Whilst PBVS estimates the position of the target after calibration, IBVS performed the control only in the image coordinate space without converting it to a 'real' coordinate system. The first method was chosen because it was the easiest to implement even though this solution requires more computation. (see below **Figure 5** & **Figure 6**)

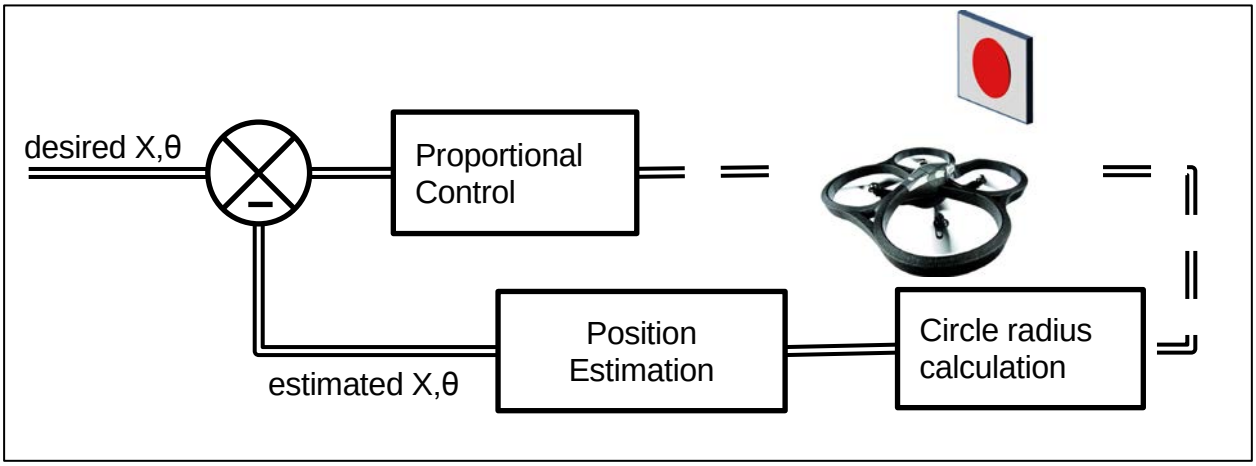

Figure 5: Position-based visual servo

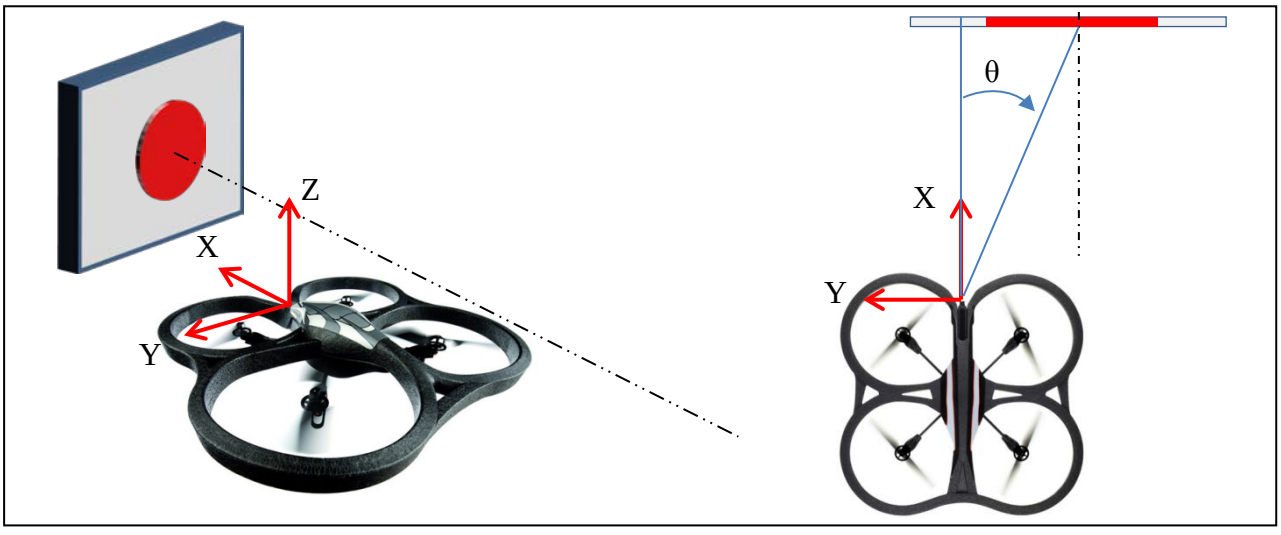

Figure 6: Position of the target with regards to the drone coordinate system

The position estimation, which determines the target position variables x, y, z and θ with regards to the drone coordinate system (**Figure 6**), was developed in the LabVIEW file

'PoseEstimation.vi' using LabVIEW Mathscript. The position estimation comprises of two stages as follows. First the distance x to the target is determined using the radius in pixels of the target and a previously determined look-up table. Secondly the y and z variables are calculated. Finally θ is calculable by trigonometry.

3.4 Sheep Detection

Before attempting to detect sheep using UAV, measurements were undertaken with a static camera, the NI Smart camera 1722C, fixed to a telehandler safety platform. Two series of photographs were taken: firstly a flock of 15 rams and secondly a single ram. To increase the contrast between the sheep and the field, the exposure time was set to more than 0.5 ms and the lens aperture opened to its widest setting.

Figure 7: image acquisition of a flock of sheep with the smart camera

After loading the photos into NI Vision Assistant software, an appropriate image processing script was developed. The most successful script used particle detection after thresholding the image. Five stages were required.

1. Image thresholding
2. Erosion to eliminate the image noise
3. Dilation to compensate for the erosion of the sheep shape during stage 2.
4. Particle filtering to eliminate all the particles whose areas is not in the interval [1000 pix2, 50000 pix2]
5. Particle analysis to determine the location of the particle centre of gravity and write it to the script output.

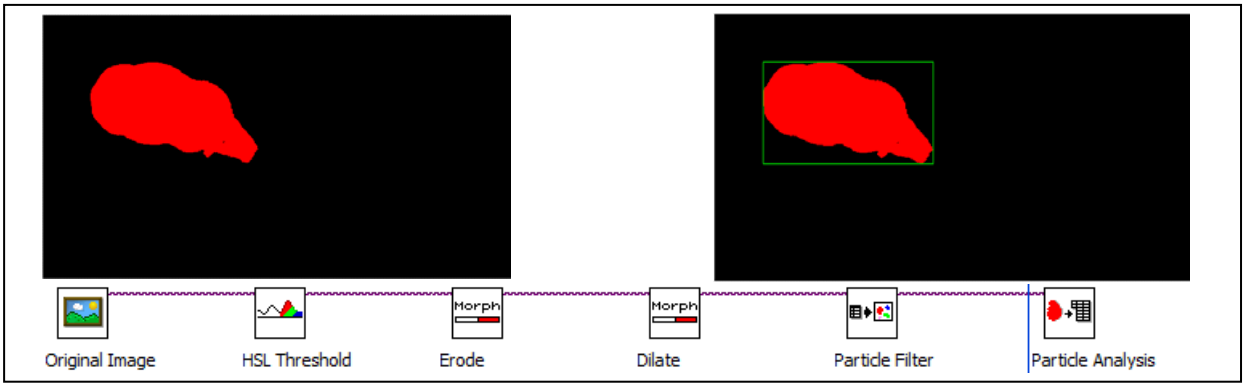

Figure 8: Sheep detection via particle analysis with National Instrument's Vision Assistant

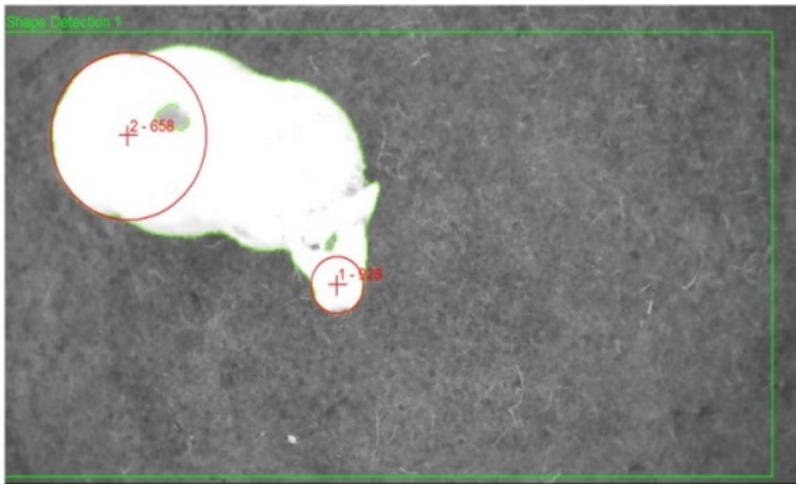

Figure 9: Sheep detection via contour analysis

Results showed that the particle detection worked quite well and was capable of detecting sheep. However, one difficulty was encountered; objects of the same colour and area as sheep are recognised as a sheep, even though they are not sheep. A number of solutions were considered to resolve this problem. One solution involved detecting the arcs of the outline of the sheep. The occluded arcs at the back of the sheep and the head are by easy to detect using an image processing algorithm (**Figure 9**). However, this technique is not as robust as particle filtering. Another solution would be to develop a more intelligent algorithm capable of detecting flexible shapes (TREIBER 2010) as all sheep are different and change shape when they turn their heads or move their ears.

4 Navigation and Control

The control directory comprises of state machines which define the sequence of the different operations of the programme. It also comprises of feedback loop control.

4.1 State machine

A Mealy automat was selected and developed in LabVIEW in order to coordinate the different actions of the drone with regards to the actual state of the drone (such as hovering or landing) and with regards to the user commands. A Mealy automat was chosen instead of a Moore one because it is easier to implement in LabVIEW. (MUETTERLEIN 2009). **Figure 10** below explains the sequence of the program. OK_0, OK_1, OK_2 and OK_3 are user events, which are triggered when the user clicks on the particular button.

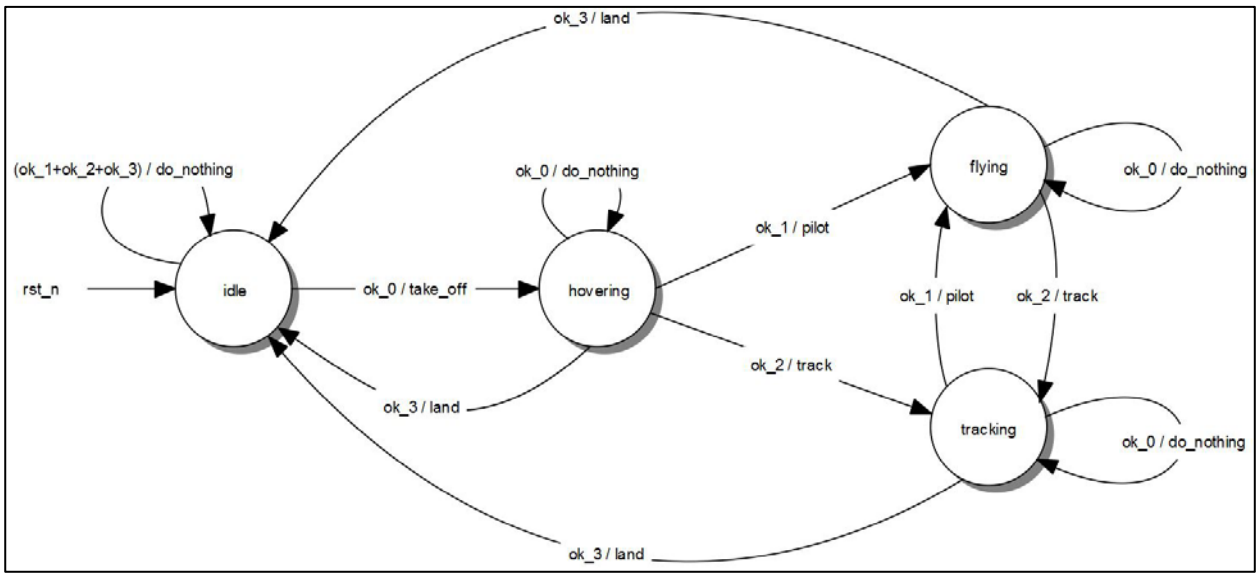

Figure 10: Design of the State Machine (realised with Qfsm)

The state graph, shown in **Figure 10** above, was programmed in LabVIEW via an array named State Event Matrix.

4.2 Tracking and Feedback loop Control

As the base station was equipped with a dual-processor, more than one operation could take place in LabVIEW at the same time. There is one requirement for this: the operations which run at the same time should be written in two different while-loops. Two feedback loops make the target tracking possible: one controls the distance in the x axis between the drone and the target, another, the angle between the x direction of the drone and the centre of the target (**Figure 11**).

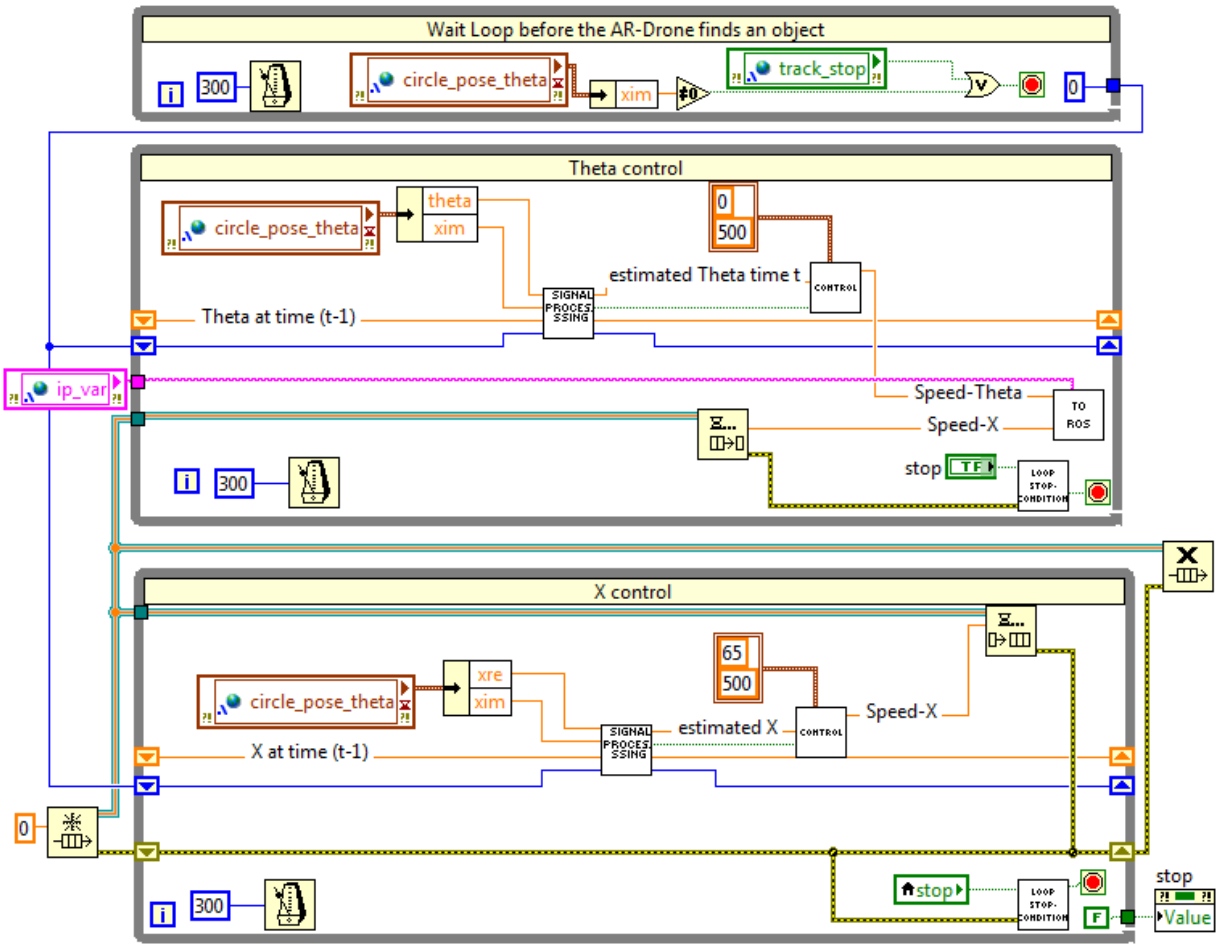

Figure 11: Tracking and feedback loops control

5 Tests

5.1 Indoor tests

Indoor testing was undertaken to test the functionality of the prototype: navigation via a joystick and target tracking. During indoor testing the AR.Drone was equipped with its indoor hull which protects the drone in the event of it hitting a wall and also makes it possible to catch the drone by hand. The results were encouraging; the AR.Drone was able to follow a defined target which had been detected previously. Whilst the tracking functionality worked, potential improvements will be required to make the tracking of objects faster. This will be necessary to track sheep.

5.2 Outdoor test with the red disk target

Figure 12: First outdoor testing weren't successful

After meeting the objectives in terms of indoor testing, trials were undertaken outside using the red disk target when the weather was not too windy. It was found that the program which worked in the sports hall did not work outside (see **Figure 12**). Further investigation of the AR.Drone documentation led to two parameters in the base station being changed and to a change in the material configuration of the experiment. In the ROS virtual machine two parameters “outdoor” and “flight without shell” had to be set to one and the AR.Drone indoor hull had to be changed for an outdoor one. Finally testing will now take place on a rugby pitch as it is tree free.

5.3 Outdoor test with sheep

After testing the functional prototype outside, tests will be undertaken to detect sheep with the camera mounted on the AR.Drone2.0. At present, these tests have not yet been undertaken. This procedure is complex for a number of reasons. Firstly, because sheep are naturally nervous animals (TULLY 2007). Secondly, separating a sheep from its flock, in order to perform the tracking test, will make it nervous. Finally authorisation from Harper Adams’ Ethics Committee has to be obtained before testing can take place using animals.

Two main measures will be taken to avoid scaring the sheep. Firstly the drone will be kept at a sufficient distance from the sheep to avoid causing it distress and, if the sheep shows any signs of stress, the experiment will be stopped. If the testing shows that sheep are not upset by the drone, the drone will be used to track an individual sheep or an entire grazing

flock of 15 rams with one target animal having a red disk painted on its back, in order to distinguish it from the others.

6 Result and discussion

This project has allowed the author to put control theory into practice, develop his programming skills and knowledge of a number of import pieces of control software and build an understanding of the use and limitations of UAVs.

The developed architecture may at first seem complicated and indirect. It is important to point out that it is the architecture for the prototyping phase of the product. The final program will be directly written in the ROS virtual machine. Clearly this architecture is not entirely suitable from a control engineering point of view. The delay is greater than if the program were directly executed in ROS. Although not optimal, this architecture has two features: the use of LabVIEW and the Vision Development Module which make prototyping quicker and accessible by people who are not programmers.

Acknowledgments

Firstly, thanks go to the Claas Foundation who makes this work possible by financing the NI Smart Camera 1722C and the LabVIEW Vision Development Module. Secondly, gratitude goes to the Bomford Trust and the Max Eyth Stiftung who provided additional funding. Special thanks should also be given to Ryan Gariepy from Clearpath Robotics who provided guidance on interfacing the Virtual Machine to LabVIEW using the MJPEG server and the ROS Toolkit; and also to Mani Monajjemi from Automomy Lab for his support and invaluable information concerning the ardrone-autonomy ROS Driver. Finally, the author is sincerely grateful to Christina Umstatter und Tony Waterhouse from Scotland's Rural College for providing him a precise understanding of the Scottish rural issue.

References

BLUME P.A. (2007): The Labview Style Book, Prentice Hall

CLEARPATHROBOTICS (2012): Overview of the LabVIEW-ROS Toolkit, 06 2012. [Online]. Available: http://files.clearpathrobotics.com/ROS%20Toolkit%20Example.pdf. [Accessed 03 2013]

CORKE P. (2013): Robotics, Vision and Control, Springer

LINZ A., RUCKELSHAUEN A. (2012): Educational robotic platform "Zero2Nine" for autonomous navigation and tracking based on imaging sensor systems, in 3rd International Conference on Machine Control & Guidance, March 27-29, 2012

MONAJJEMI M. (2012): ardrone_autonomy: a ROS Driver for ARDrone 1.0 & 2.0, July 2012. [Online]. Available: https://github.com/AutonomyLab/ardrone_autonomy. [Accessed 03 2013]

MUETTERLEIN B. (2009): Handbuch für die Programmierung mit LabVIEW, Spektrum Verlag

PITZER B.(2011): mjpeg_server, 06 2011. [Online]. Available: http://www.ros.org/wiki/mjpeg_server. [Accessed 03 2013]

QUIGLEY M., CONLEY K., GERKEY B.P., FAUST J., FOOTE T., LEIBS J., NG W.R. AND A.Y. (2009): ROS: an open-source Robot Operating System, in IEEE International Conference on Robotics and Automation (ICRA' 11), Workshop on Open Source Software

TREIBER M. (2010): An introduction to object recognition, Springer

TULLY W. (2007): Working sheep dogs, Land Links

THOMSON S., HOLLAND J., WATERHOUSE T., MORGAN-DAVIS C. (2011): Response from the Hills, 11 2011. [Online]. Available: http://www.sruc.ac.uk/info/120335/support_to_agriculture/81/2011_response_from_the_hill. [Accessed 04 2013].

WATERHOUSE T., HOLLAND J., MORGAN-DAVIS C. (2007): Decoupling and the Uplands, [Online]. Available: http://www.sruc.ac.uk/downloads/file/624/decoupling_and_the_uplands. [Accessed 04 2013]