

# Airborne Vision Sensor Detection Performance Simulation

Ali Haydar GÖKTOĞAN, Salah SUKKARIEH, David COLE, Paul THOMPSON

*ARC Centre of Excellence for Autonomous Systems*

*Australian Centre for Field Robotics*

*School of Aerospace, Mechanical, and Mechatronic Engineering*

*University of Sydney, NSW Australia*

*{agoktogan, salah, d.cole, p.thompson}@acfr.usyd.edu.au*

**Abstract.** This paper presents an implementation of sensor performance algorithms in the Real-Time Multi UAV Simulator (RMUS). The RMUS was developed to simulate missions to be performed by the Brumby Mk III UAVs at the Australian Centre for Field Robotics (ACFR). Most typical missions rely on the use of onboard mission sensors, vision in particular. It is therefore important to have an indication of the expected performance of the sensors given a particular type of mission trajectory prior to the flight. The algorithms presented allow simulation of the sensor footprint on the ground including the relative quality (pixel density) of sensor return within the footprint. The simulation aids in designing and validating trajectories for observing ground based features.

## 1. INTRODUCTION

The use of *Unmanned Aerial Vehicles* (UAVs) has many important applications, both military and civilian. Most applications require the use of onboard mission sensors (e.g. Vision sensors, Radar, etc) to observe the UAVs environment. Such information gathering tasks include reconnaissance, precision agriculture [1], surveillance [2], search and rescue [3], and mapping [4].

These tasks are related with the work conducted at the *Australian Centre for Field Robotics* (ACFR) where various mission sensor payloads have been designed and developed for the Brumby Mk III UAVs. These include *Millimetre Wave* (MMW) Radar, monochrome, colour, and *Infrared* (IR) vision sensors, and *Secondary Imaging System* (SIS) which is a vision system improved with laser range finder, mounted in various orientations within the UAV's nose cone.

For a typical UAV mission, a vehicle path is generated such that the features/targets of interest are observed by the sensors from the desired perspective. Using sensor characteristics, the simulation is able to show the location of the sensor footprint on the ground. The variations of the relative quality (pixel size) across the footprint can also be visualized over the terrain. The simulation can thus provide the expected output of the sensor which can be used to recalculate the path in order to maximize the use of a given sensor in observing a feature.

This paper is organised as follows: Section 2 introduces the RMUS environment. Section 3 describes the Brumby Mk III UAV, and Section 4 presents the vision sensor payloads used on the UAV. Section 5 describes the mathematical formulation of the simulation model to be used for the vision sensor performance calculation based on the sensor frustum. Section 6 presents results using logged data from a recent flight trial. Finally, conclusion, further works are provided.

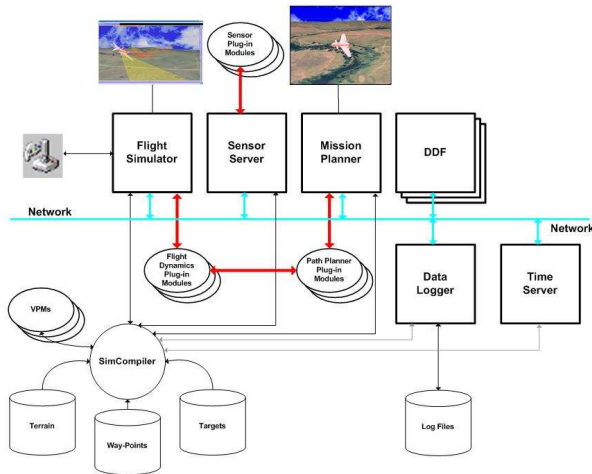
## 2. SIMULATION IN THE RMUS

Experiments with the UAVs are expensive and have associated risks. There are many avenues of failure including software and hardware, conceptual understanding, and failures in algorithm development are just a few of the many possible failure points. These pose significant burdens on the progress of the UAV projects as each flight test poses a significant cost and risk to the aircraft. Hence any new hardware or software module has to be thoroughly tested before use on a real system.

At the ACFR, a *Real-time Multi-UAV Simulator* (RMUS) has been developed to address the problems associated with real life UAV experiments [5]. In the RMUS, it is possible to simulate multiple UAV missions, and their interactions/communications with each other, the ground station, mission control and onboard systems.

The RMUS offers a comprehensive set of tools for software development, debugging, integration, validation and deployment. It also provides *Hardware-In-the-Loop* (HWIL) simulation functionality for hardware integration and both inter and intra platform communications tests. The software and hardware, once validated, are then ported directly to the UAV platforms ready for real tests. The RMUS has been used for many multi-UAV simulations [6] and also in real flight trials as a mission control system [7].

Most typical missions rely on the use of onboard mission sensors, vision in particular. It is therefore important to have an indication of the expected performance of the sensors given a particular type of mission trajectory prior to the flight. The *Vision Sensor Detection Performance Simulator* (VSDPSim) has been developed for the performance analysis of airborne vision sensors. It has been implemented in the RMUS environment.



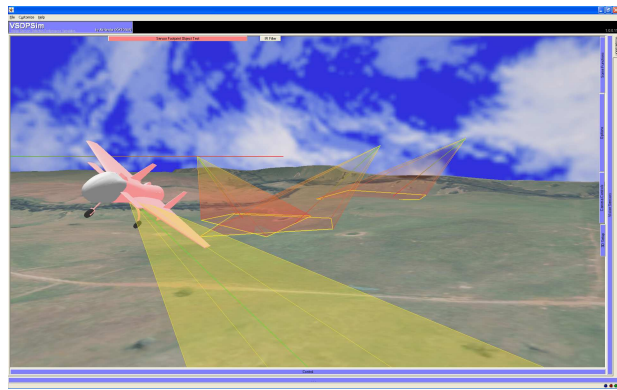
**Figure 1:** A typical RMUS configuration for the airborne *Vision Sensor Detection Performance Simulator* (VSDPSim) in a multi-UAV mission.

The modular and flexible framework of the RMUS environment allows integration of new mission scenarios, sensors, vehicle dynamics and atmospheric models to the simulation. Complex simulations can be distributed over multiple computers depending on the processing power required.

Figure 1 shows a simplified version of a typical RMUS configuration performing a VSDPSim run. The simulation has been executed in distributed manner on a small number of networked computers.

The major RMUS modules used in the VSDPSim are;

- The *Flight Simulator* (FS) provides a high-resolution, real-time display of the simulated scene consisting of the 3D terrain, UAVs and their flight paths, targets, sensor frustums. Through its graphical user interface, the simulation operator can look at the scene from a *Virtual Scene Camera* (VSC). The VSC parameters; position, orientation, and zoom factor can be changed either interactively or through the script code. The VSC can also be attached to any simulated object so that as the simulated object moves in the scene, the VSC moves along with it. This is particularly useful to generate video streams from the simulated objects' view. For example, if the VSC is attached to a vision sensor on the simulated UAV, then the operator can generate a video stream from the UAV's camera view [8]. Figure 6 shows a screen copy from the FS module running the VSDPSim in a multi-UAV coordinated sensing scenario.
- The *SimCompiler* is a simple scene description and simulation control language designed for the RMUS environment. It reads the simulation and scene description code and compiles it to run in the FS.
- The *Sensor Server* supports integration of user defined sensor models in the simulation. The simulated sensors can interact with all simulated objects in the simulation. The performance calculation algorithm which is described in Section 5 has been implemented as a plug-in module for the sensor server.



**Figure 2:** A screen image from an airborne *Vision Sensor Detection Performance Simulation* (VSDPSim), showing side looking sensor frustum.

The sensor server communicates with the FS module, and receives the UAVs position, and attitude data. This data, together with the sensor's relative position and orientation with respect to the UAV, can be used to calculate the sensor footprint.

The observations obtained from the sensor models can be sent to the *Decentralised Data Fusion* (DDF) nodes for processing. The details of the DDF process can be found in [6] [8] [9].

- The *Mission Planner* provides fully autonomous or semi-autonomous mission planning for multiple UAVs. It also allows human operators to define and generate flight paths for the UAVs to follow. Depending on the mission goal(s), different types of path planning algorithms may be implemented as plug-in modules for the mission planner.
- The *Data Logger* plays a vital role. Any data produced during the simulation can be logged for further analysis and/or replay of the simulated mission.
- The *Time Server* broadcasts time messages to the network. All of the simulation modules are time synchronized via the time server. The simulation speed can be adjusted by changing the time message frequency. This is extremely useful to simulate slower or faster than real-time simulation as well as putting break points into the simulations.
- The *CommLibX* is the framework for simulation modules to communicate with each other through virtual channels. All simulation modules communicate with each other via CommLibX. The RMUS architecture consists of a growing number of simulation modules. They are being developed by different groups on different hardware units running various operating systems. CommLibX provides hardware and operation system independent abstraction for communication in and between RMUS clusters [5].

### 3. THE BRUMBY MK III UAV

The high-fidelity dynamic model of the Brumby Mk III UAV has been developed as a plug-in module for the RMUS' flight simulator. The Brumby Mk III UAV is a sophisticated tool for *Research and Development* (R&D) projects. It has a delta wing, pusher type

propeller and conventional tricycle undercarriage suitable to be operated on variety of surfaces. It is compact for a given wing area and has a minimal component count. The pusher design allows for a clear space in the front for sensors providing an unobstructed field of view with no exhaust contamination.

The Brumby UAV offers a large payload volume, a speed range of 55-100 Knots, 2.9m wing span, a payload mass of over 13 kg and space for over 1 hour worth of fuel. The airframe is modular in construction to enable the replacement or upgrade of each component. The payload attachment system is also modular to enable easy interchange of payloads.

The Brumby UAV has been developed over the past seven years at the University of Sydney. It was not designed for high altitude, high speed or long endurance. It was designed as a test bed for various research projects.

#### 4. VISION SENSORS

The Brumby Mk III UAVs, as shown in Figure 3, are capable of carrying a variety of mission sensors. The replaceable nose cone is used to attach either a *Secondary Imaging System* (SIS) (a) or *Millimeter Wave* (MMW) radar (b) or colour and IR camera pair (c). The UAV also carries a monochrome vision camera (d) fixed to the fuselage in a downward looking configuration. Similarly, the SIS which is a vision system enhanced with a laser range finder is also positioned in the nose cone looking downwards.

The current colour vision sensor consists of a digital Sony XCD-X710CR CCD camera with 1024x768 pixel spatial resolution, interfaced to Pentium based PC104+ computer with IEEE 1384 interface.

The IR vision sensor consists of an internally cooled Inframetrics Infracam camera with detection spectrum of 3-5 $\mu$ m and 255x255 pixel spatial resolution. It has composite video output and is connected to a frame grabber on board on a separate computer.

Both the captured colour and the IR video frames are being recorded to hard disks. The video frames are stamped with the system time and the UAV navigation data. This data can later be analysed to verify the fidelity and accuracy of the simulation system.

As shown in Figure 3-(c), the colour and IR cameras are mounted into the nose cone in a sideward looking manner instead of traditional forward looking form. This orientation for the vision sensors is particularly useful to gather 360<sup>0</sup> data of a target as the UAV orbits around it with a constant bank angle.

In order to maximize the sensor performance, the path planning modules optimise the bank angle, flight altitude and the flight velocity within the safe flight envelope of the UAV dynamics.



**Figure 3:** The replaceable nose cone of the Brumby Mk III UAVs is used to attach either an (a) SIS or (b) MMW radar or (c) colour and IR camera, or (d) a monochrome vision camera is fixed to the fuselage in down looking configuration.

### 5. SENSOR PERFORMANCE MODEL

All missions simulated in the RMUS are focused on observations made by the onboard sensors. Therefore it is vital to be able to simulate the sensor return within a simulation. This section describes how the sensor performance is explored through constructing a sensor footprint. Also variations in the quality of the sensor performance over the footprint are measured. In the RMUS environment the sensor performance model is implemented as plug-in module for the sensor server. In the sensor server, the sensor performance model can be linked with any simulated vision sensor with rectangular footprint.

#### 5.1 Sensor Footprint

Calculating the location of the sensor footprint on the ground is necessary when simulating the UAV missions. It gives an indication of the areas on the ground which are being observed. The footprint calculation requires performing coordinate transformations from the *sensor* frame to the *earth-fixed inertial* frame. A brief description of notations used and coordinate transformations follows.

The position  $(x,y,z)$  of a coordinate frame  $j$  with respect to another  $i$  is given by  $\mathbf{P}_j^i$ . The orientation; yaw ( $\psi$ ), pitch ( $\theta$ ), roll ( $\phi$ ) of the frame  $j$  with respect to frame  $i$  is represented by the transformation matrix  $\mathbf{C}_j^i$  from  $i$  to  $j$ . The coordinate transformation matrix is calculated as shown in equation (1), where  $S_{(\cdot)}$ , and  $C_{(\cdot)}$  are  $\sin(\cdot)$  and  $\cos(\cdot)$  respectively. Equation (2) shows the reverse transformation from  $j$  to  $i$ . Equation (3) shows how the position of a point  $k$  with respect to frame  $i$  is calculated from its position with respect to frame  $j$ .

$$\mathbf{C}_j^i = \begin{bmatrix} C_\theta C_\psi & C_\theta S_\psi & -S_\theta \\ -C_\phi S_\psi + S_\phi S_\theta C_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & S_\phi S_\theta \\ S_\phi S_\psi + C_\phi S_\theta C_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi & C_\phi C_\theta \end{bmatrix} \quad (1)$$

$$\mathbf{C}_i^j = \mathbf{C}_j^{iT} \quad (2)$$

$$\mathbf{P}_k^i = \mathbf{C}_i^j \mathbf{P}_k^j \quad (3)$$

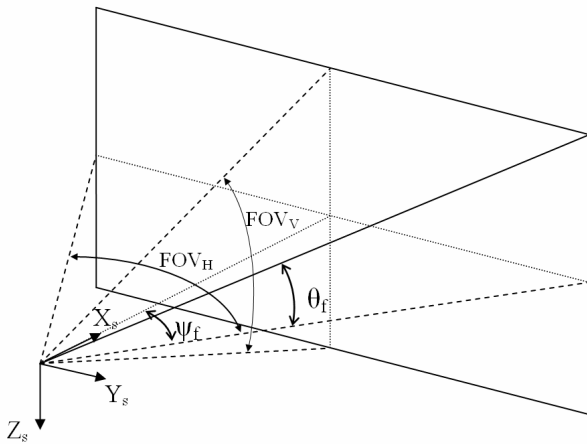
The location of the sensor footprint depends on four sets of variables:

1. The *field of view* characteristics of the sensor. Horizontal and vertical fields of view are denoted by  $FOV_H$  and  $FOV_V$  respectively.
2. The location and orientation of the *sensor* ( $s$ ) frame relative to the UAV *body* ( $b$ ) frame ( $\mathbf{P}_s^b$  and  $\mathbf{C}_s^b$ ).
3. The location and orientation of the UAV *body* frame relative to the *earth-fixed* ( $e$ ) coordinate frame ( $\mathbf{P}_b^e$  and  $\mathbf{C}_b^e$ ).
4. The Z location of the ground plane in the inertial frame ( $Z_{Ground}$ ).

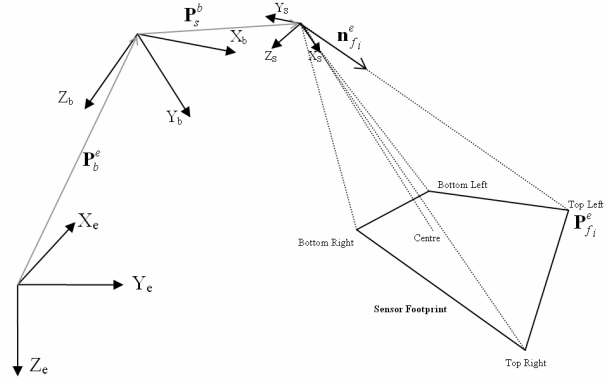
The field of view of the sensor defines the transformation matrix  $\mathbf{C}_{f_i}^s$  from the *sensor* frame to the unit vector  $\mathbf{n}_{f_i}^e$  pointing along the sensor corner ( $f_i$ , for each corner  $i$ ). Figure 4 shows the fields of view and their relation to the angular rotations required to transform from the sensor frame to a corner. Combinations of  $\pm\psi_f$  yaw in equation (4),  $\pm\theta_f$  pitch in equation (5) and zero roll give transformation matrices for all four corners.

$$\psi_f = \frac{FOV_H}{2} \quad (4)$$

$$\theta_f = \arctan\left(\tan\left(\frac{FOV_V}{2}\right)\cos\psi_f\right) \quad (5)$$



**Figure 4:** The horizontal FOV is measured in the XY plane of the sensor coordinate system. The vertical field of view is measured in XZ plane. The angles  $\psi_f$  and  $\theta_f$  transform from the centre of the sensor frame to a corner of the FOV.



**Figure 5:** The footprint is calculated by projecting the four unit vectors  $\mathbf{n}_{f_i}^e$  corresponding to each of the corners where the earth fixed inertial frame, the UAV body frame, and the sensor frame are denoted by  $e$ ,  $b$ , and  $s$  respectively.

Figure 5 shows the different coordinate frames and the sensor footprint projected on the ground. Using the above transformations the value of the unit vector  $\mathbf{n}_{f_i}^e$  can be calculated as in equation (6). The origin of all unit vectors is the position of the sensor in the *earth* frame (7).

$$\mathbf{n}_{f_i}^e = \mathbf{C}_e^b \times \mathbf{C}_b^s \times \mathbf{C}_{s_i}^f \times \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad (6)$$

$$\mathbf{P}_s^e = \mathbf{P}_b^e + \mathbf{C}_e^b \mathbf{P}_s^b \quad (7)$$

Calculation of the intersection with the ground for each unit vector, involves solving the parametric line equations (9) (10) (11) simultaneously given equation (8) which defines the ground XY plane.

$$z = Z_{Ground} \quad (8)$$

$$x = x_0 + lt \quad (9)$$

$$y = y_0 + mt \quad (10)$$

$$z = z_0 + nt \quad (11)$$

where,

$$\mathbf{P}_s^e = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \quad (12)$$

$$\mathbf{n}_f^e = \begin{bmatrix} l \\ m \\ n \end{bmatrix} \quad (13)$$

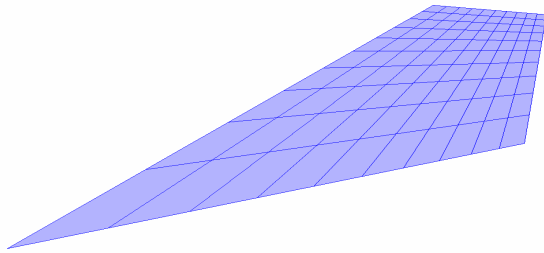
This process means that all types of vision sensors with rectangular footprints can be modelled. This also includes sensors which can pan, tilt and zoom independently of the UAV.

## 5.2 Sensor Performance Measure

Calculating the sensor footprint defines a region on the ground which the sensor is able to see. It does not give any indication of the quality of the sensor information obtained at each point in the region. This is important as plotting the sensor footprint on its own can often be misleading. In the cases where we desire to point the sensor at a feature or region on the ground, the aim is not only to have the feature within the sensor footprint but within a region of the sensor footprint which has high pixel coverage. Pixel density is used as a performance measure to show the pixel coverage over the footprint.

In order to visualise this variation in sensor quality over the footprint in simulation, the footprint is divided into grids of equal pixel size. Therefore the larger the grid appears on the ground the smaller the pixel density is in that region. Hence the quality of the information obtained by the sensor is less than from an area of high pixel density.

The process of calculating the grid is similar to that described in Section 5.1. The fields of view are divided angularly into  $n$  number of sections each containing an equal number of pixels. This is done for each of the four sides of the sensor footprint. Unit vectors are projected as before, the intersections with the ground plane are joined to produce the grid as shown in Figure 6.



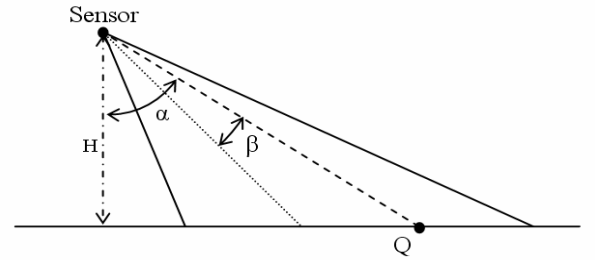
**Figure 6:** Sensor footprint on the ground as viewed from above. The grid areas represent regions covered by an equal number of pixels.

The pixel density (pixels/m<sup>2</sup>) is used as a quantitative measure of the quality of the sensor return from a particular point within the footprint. As expressed in equation (14), the uni-directional pixel density at any point  $Q$ , within the footprint, is dependent on the position of the sensor  $P_s^e$  and the angular size of the

pixels of the sensor  $\gamma$ . The angular size of the pixel is given in equation (15). The pixel angular size depends on the focal length of the sensor ( $\delta$ , measured in pixels) and varies depending on the offset ( $\beta$ ) of the feature from the centre of the frame as shown in Figure 7. The angle alpha  $\alpha$  is the angle between the unit vector from the sensor to  $Q$ , and a normal to the ground plane,  $H$  is the height of the sensor above the plane.

$$\sigma = \frac{1}{H \sec^2(\alpha) \gamma} \quad (14)$$

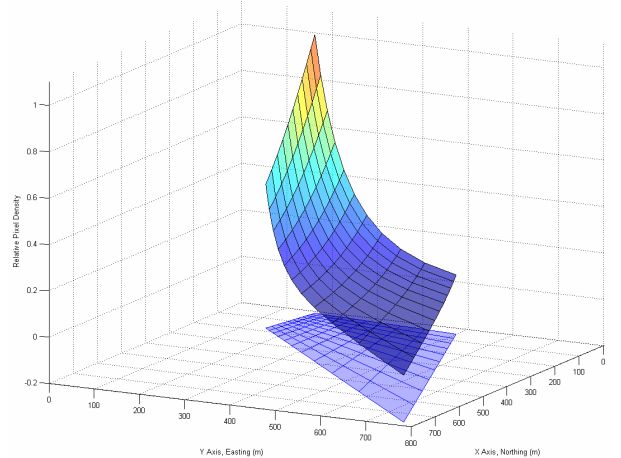
$$\gamma = \frac{1}{\delta \sec^2 \beta} \quad (15)$$



**Figure 7:** Angles required for calculation of pixel density at  $Q$  along one direction of the sensor footprint

Calculation of the pixel density has been presented here for a two dimensional case. In three dimensions the same equations are used to calculate both the horizontal and vertical pixel densities at each point, which are multiplied to give the pixels/m<sup>2</sup> measure.

Pixel density is calculated for every point on the grid. A relative measure of quality can be obtained by dividing by the largest pixel density within the footprint. This leads to a pixel density distribution as shown in Figure 8. The same process can be used to calculate the pixel density at the exact location of a feature on the ground throughout simulation.



**Figure 8:** Graph showing the relative pixel density at each point on the sensor footprint below.

The method described above enables UAV trajectories to be designed such that they make best use of a given sensor in order to observe a particular region or feature. Such information is valuable in designing trajectory types for a current project which involves investigating detection and classification of ground features. It is limited by the fact that it assumes a flat ground. This is often not the case; however, variations in topology are less relevant when viewed from the large heights flown by UAVs.

## 6. FLIGHT TRIAL

We have successfully performed a remotely piloted flight test with the Brumby Mk III UAV on Feb 2005 for the purpose of logging colour and IR vision data. Figure 9 shows (a) the UAV in flight and also (b) colour and (c) IR image frames. Each image frame on the logged data is time stamped and associated with the navigation solution (position, velocity and attitude) data.



**Figure 9:** (a) The Brumby Mk III UAV in flight. It is equipped with a colour and an IR vision sensor. A (b) colour and (c) IR image frames showing different sections of the runway.

The logged vision data from the real flight test are being compared with the simulated vision sensor performance data to verify the fidelity and accuracy of the simulation models and the simulation system. Initial results are promising. The VSDPSim will be used in mission planning for the next flight trials in which UAV will autonomously fly predefined paths and orbit around designated ground feature of interests.

## 7. CONCLUSION AND FURTHER WORKS

This paper has presented the RMUS environment, the Brumby Mk III UAV and its vision sensors. The mathematical foundation of the vision sensor performance calculation based on the sensor frustum is provided. The sensor performance model is used on the simulation of countless different mission scenarios.

The work is now being extended to the development of multi-objective optimisation strategies for automatic path planning to maximize information gain from the vision sensor.

Currently released versions of the RMUS do not fully comply with the *High Level Architecture* (HLA). We are working toward the full compliance, and once

achieved, it will be possible to link the RMUS with the existing C2 systems and simulators to evaluate UAV deployment and tactics.

## Acknowledgment

This work is supported by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

## REFERENCES

1. Stanley R. Herwitz, Lee F. Johnson, John C. Arvesen, Robert G. Higgins, Joseph G. Leung and Stephen E. Dunagan, (2002), "Precision Agriculture as a Commercial Application for Solar-Powered Unmanned Air Vehicles", *1st AIAA UAV Conf, AIAA 2002-3404*
2. Michael, P. K., Tsourveloudis, N. C. and Valavanis, K. P. (2003), "A UAV Based Automated Airborne Surveillance System", *CD Proceedings of the 11th Mediterranean Conference on Control and Automation*
3. Bourgault, F., Göktoğan, A.H., Furukawa, T., Durrant-Whyte, H.F., (2004), "Coordinated Search for a Lost Target in a Bayesian World", *Advanced Robotics, Special Issue on Selected Papers from IROS, (18):pp. 979-1000*
4. Eisenbeiss, H., (2004), "A Mini Unmanned Aerial Vehicle (UAV): System Overview and Image Acquisition," *International Workshop on Processing and Visualization Using High-Resolution Imagery*
5. Göktoğan, A.H., Ridley, M., Nettleton, E., Sukkarieh, S., (2003), "Real Time Multi-UAV Simulator", *IEEE International Conference on Robotics and Automation (ICRA'03)*, pp: 2720-2726
6. Sukkarieh, S., Yelland B., Durrant-Whyte, H., Belton, B., Dawkins, R., Riseborough, P., Stuart, O., Sutcliffe, J., Vethecan, J., Wishart, S., Gibbens, P., Göktoğan, A. H., Grocholsky, B., Koch, R., Nettleton, E., Randle, J., Willis, K., and Wong, E., (2001), "Decentralised Data Fusion Using Multiple UAVs - The ANSER Project", *3rd International Conference on Field and Service Robotics*
7. Sukkarieh, S., Göktoğan, A. H., Kim, J.-H., Nettleton, E., Randle, J., Ridley, M., Wishart, S. and Durrant-Whyte, H. (2002), "Cooperative Data Fusion and Control Amongst Multiple Uninhabited Air Vehicles", *ISER'02, 8th International Symposium on Experimental Robotics*
8. Göktoğan, A.H., Sukkarieh, S., (2005), "A Augmented Reality System for Multi-AUV Missions". *To appear in SimTect'05*
9. Nettleton, E., Ridley, M., Sukkarieh, S., Göktoğan, A.H., Durrant-Whyte, H., (2004), "Implementation of a Decentralised Sensing Network aboard Multiple UAVs", *Telecommunication Systems Special Issue: Wireless Sensor Networks, (26:2-4)*, pp:253-284