SUAAVE: Combining Aerial Robots and Wireless Networking

Stephen Cameron[o] Stephen Hailes[l] Simon Julier[l] Sally McClean[u]
Gerry Parr[u] Niki Trigoni[o] Mohamed Ahmed[l] Graeme McPhillips[l]
Renzo De Nardi[l] Julia Nie[u] Andrew Symington[o] Luke Teacy[u]
Sonia Waharte[o]

[o]University of Oxford [l]University College London [u]University of Ulster

Final Draft, February 2010

Abstract

The SUAAVE project is funded by EPSRC under the WINES wireless networking initiative to consider issues of multiple aerial vehicles communicating and collaborating in performing tasks, and involves teams from University College London, University of Ulster, and the University of Oxford. The focus of SUAAVE lies in the creation and control of swarms of UAVs that are individually autonomous (i.e not under the direct realtime control of a human) but that collaboratively self-organise: to sense the environment in the most efficient way possible; to respond to node failures; and to report their findings to a base station on the ground. As the focus is on developing algorithms, rather than rugged hardware, we are using small off-the-shelf electric quad-rotor vehicles with payloads of a few hundred grams. Although the technology is not tied to a particular scenario, we are basing our research on a search-and-rescue scenario, whereby a swarm of vehicles are searching for a lost or injured person.

Biographies

Dr Cameron's research area is in spatial reasoning and robotics. **Prof Hailes** is the PI for SUAAVE, and his research interests cover all aspects of mobile systems and security. **Dr Julier** is interested in the problem of developing, maintaining and conveying situation awareness, and especially in problems which are distributed and mobile. **Prof McClean**'s background is in statistical modelling, with particular interests in ambient intelligence, and optimisation. **Prof Parr** has many interests, but especially secure telecommunications. **Dr Trigoni**'s focus is on sensor data networks. **Mr Ahmed** is a research student specialising in telecommunications. **Mr McPhillips** and **Renzo De Nardi** are responsible for keeping our machines aloft, and **Dr Nie** for its safety. **Mr Symington** is a research student working in the area of mobile sensor networks. **Dr Teacy** works on trust and distributed reasoning under uncertainty. **Dr Waharte**'s interests include distributed algorithms in a wireless environment.

Contact Details

Dr Stephen Cameron, Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD. (01865) 273850, `stephen.cameron@comlab.ox.ac.uk`.

1 Introduction

The SUAAVE project is funded by EPSRC under the WINES wireless networking initiative to consider issues of multiple aerial vehicles communicating and collaborating in performing tasks, and involves teams from University College London, University of Ulster, and the University of Oxford. The project runs from late 2008–2012, with focus on the creation and control of swarms of UAVs that are individually autonomous (i.e., not under the direct realtime control of a human) but that collaboratively self-organise: to sense the environment in the most efficient way possible; to respond to node failures; and to report their findings to a base station on the ground.

The novelty of these mobile sensor systems is that their movement is controlled by fully autonomous tasking algorithms with two important objectives: first, to increase sensing coverage to rapidly identify targets; and, second, to maintain network connectivity to enable real-time communication between UAVs and ground-based crews. The project has four main scientific themes: (i) wireless networking as applied in a controllable free-space transmission environment with three free directions in which UAVs can move; (ii) control theory as applied to aerial vehicles, with the intention of creating truly autonomous agents that can be tasked but do not need a man-in-the-loop control in real time to operate and communicate; (iii) artificial intelligence and optimisation theory as applied to a real search problem; (iv) data fusion from multiple, possibly heterogeneous airborne sensors as applied to construct and present accurate information to situation commanders. The main experiments will be based on a simplified search-and-rescue scenario, with the idea that the UAVs will cooperate to find a ‘victim’ in an unstructured outdoor setting.

2 Hardware & Software

The hardware platforms we are using are off-the-shelf, electrically-powered quad-rotors; a number of such platforms have become commercially available over the last few years with similar characteristics and we are purchasing a number from Ascending Technologies (`www.asctec.de`). Such a vehicle has the advantage of being relatively easy to prepare, control and fly and has a payload of a few hundred grams. However, the disadvantage over petrol-powered machines being their endurance — typically a few tens of minutes at high load. However as an experimental platform long endurance is not a priority for us.

The original flying machine has the ability to reach autonomously GPS waypoints specified by the user but does not provide the computation and communication abilities needed for our research. We redesigned the central structure of the quadrotor to provide sufficient room for our computer board. As visible in Figure 1 the design is based on a central carbon fibre foam core plate on top of which is mounted the original UAV electronics. This hosts the inertial sensors[1] and the two ARM7 processors that run the state estimation and the FCS (flight control system) algorithms.

Under the central plate we installed a 1.6 GHz Intel Atom board[2] with 1GB of ram, a 16GB SSD, two 802.11a/b/g/n wireless cards and the necessary power stabilization circuitry. A 1.3Mpix USB camera[3] and the 2100mAh battery are mounted outside the bottom enclosure. The on-board PC and the FCS are connected through a serial link to exchange commands and flight data. The bottom plas-

---

[1] The sensor suite include three gyros, a triaxial accelerometer, three magnetic sensors and a pressure sensor, in addition to a GPS receiver.

[2] MSM200X manufactured by Digital-Logic AG.

[3] Chameleon manufactured by Point Grey Research, Inc.

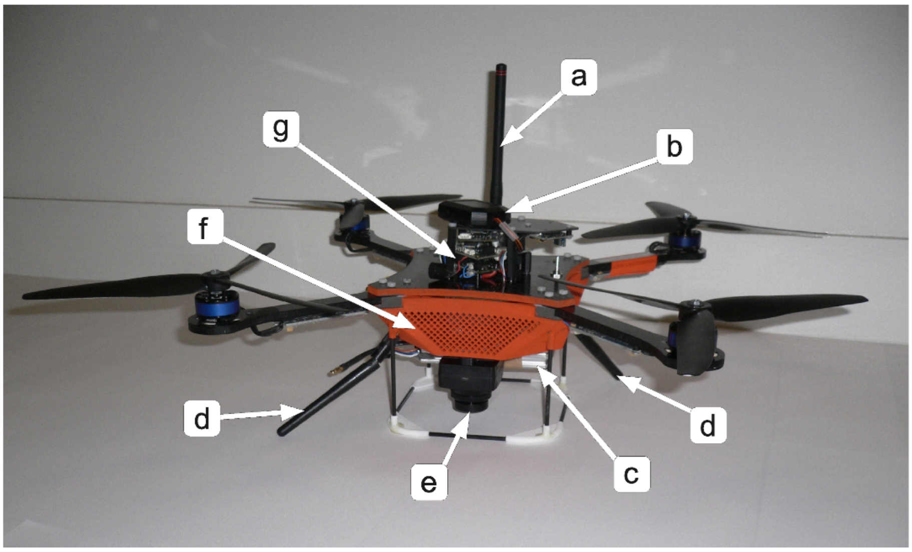

Figure 1: UAV platform: RF safety module (a); GPS (b); battery (c); wifi antennas (d); camera (e); base enclosure containing the PC (f); flight control system (g).

tic enclosure was realized using a 3D printing process and has a grid structure in order to be light and allow sufficient airflow for the internal electronics. Empirical testing revealed that the computer board generates EMF radiation at levels and frequencies that heavily interferes with the GPS unit placed at the top of the UAV. It was therefore necessary to coat the bottom enclosure with multiple layers of conductive paint in order to ensure the necessary shielding.

After the modification the total weight of the platform was raised to 850g; to compensate the original propellers were substituted with larger, stiffer and more efficient 9 inch three blade propellers which provide more lift. The retrofitted platform has currently a flight endurance of about 10 minutes with PC and camera turned on.

Safety is a primary aspect in our project especially when testing novel control and coordination strategies; for this reason we equipped our UAV with a custom designed RF safety module completely independent from the onboard computer. This module automatically lands the vehicle when out of range from the base station, but also allows to remotely land the UAV or even to cut off the power to the rotors in the eventuality of the on-board PC becoming unresponsive. The overall system architecture is shown in Figure 2.

2.1 Software architecure

As mentioned in the previous section, there are several microprocessors on board the UAV, some of which provide autopilot functionality while others ensure the safety of the platform. However, we strictly do not run any project code directly on these microprocessors, as interfering with their normal operation might cause hardware failure, which is a safety risk to both the platform and the researchers. In stead, the code runs on the Intel Atom board, which has significantly more computational and memory resources. This board has a single serial link that connects it to the autopilot board, permitting us to send control

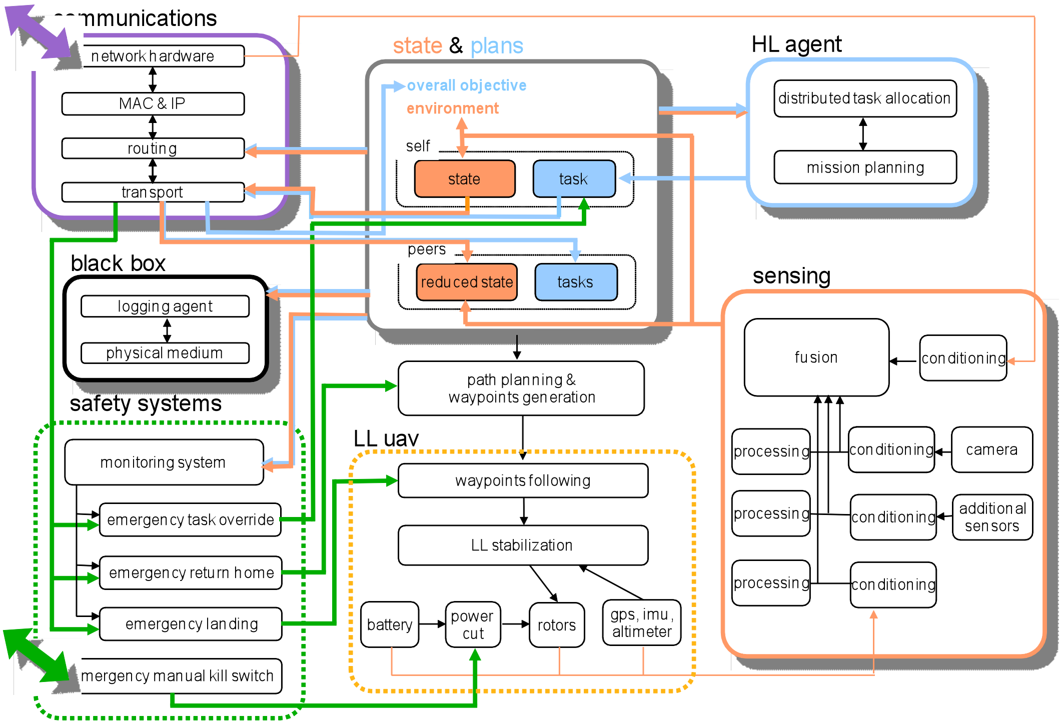

Figure 2: Overall System Architecture

instructions and receive low level sensor data.

The software architecture that we chose was centred around the concept of modularity. Our goal was to have many distinct modules running as separate processes, each being responsible for a single operational component of the system. Not only does this make collaborative project development significantly easier, but it also makes the system robust to failure. For example, if the high-level navigation code terminates unexpectedly, the UAV is still capable of performing a software landing since the remainder of the system remains operational. Our base system contains the following base modules, which are core to the reliable operation of the UAV:

1. **Control** Arbitrates communication with the autopilot and safety systems over serial connections.
2. **Safety** Monitors the messages sent between all system modules. From these messages it infers the system state. In the case where the system switches to an undesirable state, it assumes control and instructs the Control module to fly to a safe zone and land.
3. **Black-box** - Records all inter-module communication for the purposes of debugging.
4. **WaypointFollower** Parses a list of GPS waypoints and communicates with the control module to have the UAV fly from waypoint to waypoint, idling for a specified period.

Although the modules are operationally distinct, they must be able to communicate with each other. It is clear that, for example, the WaypointFollower module must communicate periodically with the Control module to issue actuation instructions. This implies that all modules must subscribe to some messaging-oriented (MOM) middleware that provides a mechanism for inter-process communication. A number of MOM frameworks already exist, such as RCS[4], The MOOS[5] and MQ4CPP[6]. Our goal is to keep the system extensible as possible, so we created an implementation-agnostic interface for our software modules that bind it to the MOM, while also defining a strict message format for modules to adopt. This allows us to switch to another MOM when the need arises without changing any module code.

For the time being we have chosen to use The MOOS, as it is straightforward to use and appears to have been designed with robotic applications in mind. The MOOS is based on a publish/subscribe system that operates over a TCP/IP network. When a module (MOOSApp) initialises it first binds to a specific messaging database (MOOSDB) and requests to be notified when certain message types are received by the MOOSDB. As such, the messaging service operates in a star-like topology, which is shown in Figure 3.

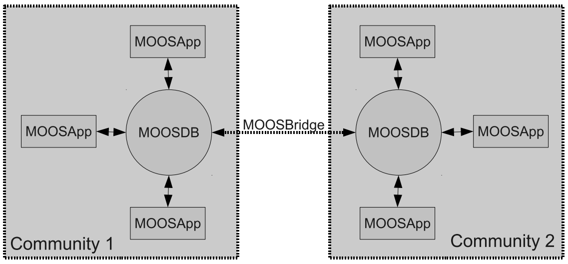

Figure 3: The MOOS architecture

In practice, each hardware unit in the network (either a UAV or a ground station) runs a single MOOSDB. It is clear that we cannot use one MOOSDB for the entire network, since this makes the fundamental assumption of perfect network connectivity. This is seldom true for wireless networks and would cause the messaging in disconnected hardware units to break down. The MOOS also provides a bridging mechanism (MOOSBridge) that allows multiple MOOSDBs to communicate with each other and, hence, forward messages between modules running on separate hardware units. However, this works on a broadcast-like mechanism, which is likely to scale poorly in multi-hop wireless networks.

3 Communications

The UAV platform's agility and ease of deployment has the potential to turn them into mobile and robust communication platforms. While a number of papers have studied the problems of network connectivity maintenance and quality of service, including [BRS04, BPDKHO09, CDB04, HSL06, PBGY08, RZ07], they have often been theoretical in their approach to the problem space, and make stringent assumptions about the operating environment or the state of the network. For example, common assumptions include accurate channel models and symmetric links, known workloads, static locations or simplistic movement models and so forth. Though these assumptions enable us reduce the complexity of the systems under investigation and provide tractable solutions that focus on specific questions, their results cannot be taken and applied directly to real operational platforms.

The SUAAVE project aims to help bridge the gap between theory and practise and extend the state of the art in two ways. Firstly, we will perform a series of measurements targeted at:

- Generating 802.11 performance data

---

[4] http://www.isd.mel.nist.gov/projects/rcs/

[5] http://www.robots.ox.ac.uk/ mobile

[6] http://www.sixtyfourbit.org/mq4cpp.htm

based on in-air measurements and real workloads.

- Devising new models to characterise the performance of 802.11 on UAV, subject to scenario details.
- Comparing proposed models of 802.11 characteristics and models with real data and updating them.

And secondly, the application of Ad-Hoc and Mesh networking to provide a communication platform for UAV swarms, as opposed to the traditional centralised communication models. In particular, this involves:

- Ruggedising existing Ad-Hoc and Mesh protocols which often make stringent demands on the mobility and link failure statistics that can be associated with nodes, to function in this space.
- Applying the derived measurement models to guide the behaviour of UAVs when performing network centric tasks.
- Developping new algorithms to support QoS constrained services, such as sensor data collection and video streaming.

4 Control

We require an integrated approach, where by each UAV takes into account the limitations of their resources and the current state of their peers when pursuing their objectives. For instance, a UAV should not blindly follow a path toward its target, if that path takes it through a prohibited flight zone, or leaves it with insufficient battery power to return to its home base. Other examples could be restrictions or limits on the amount of on-board memory to capture locally acquired sensor data when communications are not available. Instead, it should consider alternative actions, such as giving up its objective or asking for assistance from other UAVs. For this reason, it is essential that any such decision process should operate within the bounds and constraints of a wider safety management protocol (SMP) for ensuring none of the risks mentioned in the previous section lead to real-life disasters. Moreover, for such an SMP to be verifiably implemented in a safe and reliable manner, it must eliminate these risks in the simplest way possible, without placing undue restrictions on the high level decision processes that we wish to develop. We propose a basic safety architecture for the SUAAVE project, to mitigate the risks associated with operating UAVs in, or near, publicly accessible areas. In particular, we wish to avoid the risks of personal injury (both the UAV operators and the general public), damage to property through collisions, and any legal consequences due to unplanned landing of UAVs on private property.

4.1 Safety Management Protocol (SMP) Function

Phases of Operation To ensure safe flight, each UAV should perform a number of self-diagnostics and safety checks, not only at the start of each mission, but also periodically during normal flight. Moreover, should any of these checks fail, it is essential that we have clear procedures in place to either recover from failure, or to abort flight in the safest possible way. With this in mind, we propose that, from the moment it is switched on until the end of its mission, each UAV should move through five phases of operation:

1. Pre-flight bootstrap, during which the UAV performs initial diagnostic checks before taking to the air;
2. In-flight self diagnostics, where by the UAV makes further checks that must be performed during flight;
3. Operation, in which the UAV actively

pursues its high-level goals;

4. Recovery, in which the UAV attempts to recover from any detected error; and
5. Abort, through which the UAV attempts to land as safely as possible, following and unrecoverable error.

The conditions that lead to transitions between these phases are illustrated by the state diagram in Figure 4, and in more detail by the activity diagram in Figure 5. In particular, a UAV can only reach the operation phase if all initial checks have completed successfully, and will only remain there so long as ongoing self-diagnostics can verify that the system is functioning normally. If this is not case, the UAV can either attempt to recover from failure, or abort its mission entirely.

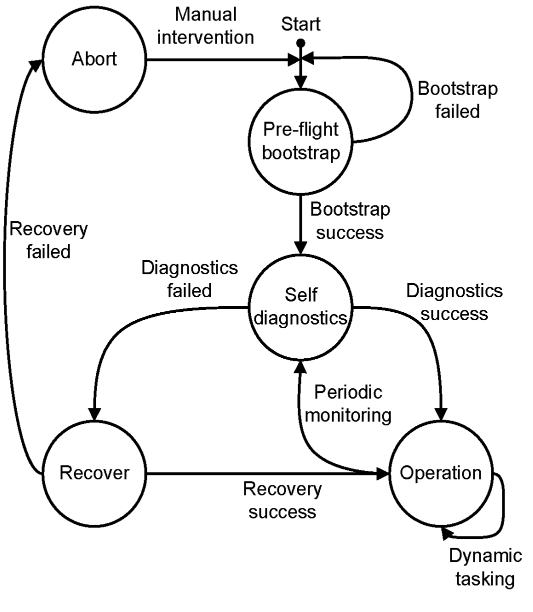

Figure 4: State transition diagram for the phases of operation

The following subsections describe each of these phases in more detail.

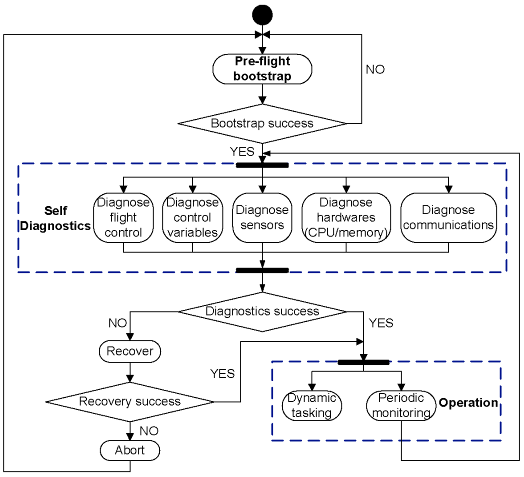

Figure 5: Activity diagram for the phases of operation

Pre-Flight Bootstrap When a UAV is first switched on, it should ensure that all resources necessary for safe flight are present and in working condition. As far as possible, this should be done before the craft becomes airborne. In particular, while on the ground, each UAV should ensure that

- A communication link can be established with a manned base station, possibly using a routing protocol and other UAVs;
- The GPS location system is active and reporting that the UAV is located at its designated home; and
- All onboard sensors are available and can respond to control signals; Only if the above checks have been completed successfully, should the UAV proceed to the following phases of operation. Otherwise, the only advisable course of action is the abort the mission immediately and seek human intervention.

In-Flight Self Diagnostics Once the above checks have been passed, and assuming the UAV has been assigned a task that requires flight, it should perform any remaining diagnostics that must be performed in flight.

Specifically, to ensure that the GPS system and helicopter rotators are operating normally, we propose that the following three steps are performed in order[7].

1. The UAV should raise to a height of 1–2 metres of the ground and hold its position for 45–120 seconds. Where GPS is available this should allow GPS Lock. The location should be verified against the reported GPS location, and ideally some other sensor. For example, if a set of red dots are placed on the ground, it should be possible to verify location using onboard cameras.
2. The UAV should attempt to perform a full rotation, and verify this ability using onboard sensors.
3. The UAV should move a set distance in 6 directions: up, down, north, south, east and west. Again, success should be verified using the GPS, and other on board sensors.

In addition to these tests, many of the diagnostics performed in the pre-flight bootstrap could be repeated periodically during flight. Together, these checks should ensure that a complete set of system control and status variables are continually monitored to ensure safe operation. For example, these could include flight and other control variables, communication links, sensor, CPU and other hardware resources (see Figure 5).

Operation and Monitoring Having perform all initial checks successfully a UAV can then proceed with the mission assigned to it. With reference to the software architecture described, this means that the safety layer and other APIs can dynamically accept tasks from the application layer, while continuing to monitor the status of critical resources (see Figure 5). In general, these task requests could take a variety of forms, including communication with a base station or other UAVs, requests for resource status information or to perform specific manoeuvres. The precise form that requests from the application layer may take will depend on both the applications requirements, and the ease with which their safety can be verified. For example, the easiest way to allow the application layer to specify movement is through a stream of GPS way-points. For many high-level decision processes, this level of detail may be sufficient, and it is also straightforward to verify such requests against the SMP. It is also worth mentioning that, as far as possible, the application layer should use available status information to avoid making decisions that would lead to breaches of the SMP. In any event, it is during this phase of operation that the safety layer should guard against risky operation, by denying requests from the application layer that would violate the SFP. Most importantly, this means that requests to move to a way point outside the safe flight zone should be denied, along with any course that risks collision with known objects or other UAV flight paths. Requests to move beyond the range of safe landing zones (given current battery life) are also prime candidates for service denial. In addition to denial of service requests, further precautions could be taken by periodically repeating some of the checks from the previous phases. The precise timing of each check could be pre-specified by configuration parameters, or could be a matter of some more detailed investigation. In any case, choosing the frequency of each check is a trade-off between its cost in terms of resources used, and the benefit in terms of risk minimisation.

---

[7] One possible problem with these proposed steps is that the GPS may also play an active role in directing these maneuvers. If this is the case, then checking the movements for consistency with the GPS may have limited value.

Recovery and Abort Sequence In the event that something does go wrong (some essential resource becomes inoperative, or the application layer becomes unresponsive) then a UAV may attempt to recover from the failure, or execute an abort sequence to land the UAV as safely as possible. For example, recovery or abort procedures should be initiated in the following scenarios:

1. The UAV looses its communication link to a manned base station[8]
2. The application layer crashes or becomes unresponsive
3. The UAV's battery level becomes dangerously low
4. The UAV receives a manual abort signal from a manned base station
5. The UAV CPU develops unrecoverable errors

Precisely which, if any, such failures can be recovered from, without the need to abort operation, is up for debate. However, the most likely candidates involve communication. For instance, if a constant communication link is deemed necessary (for example, to receive a manual abort signal) then the UAV may move in the direction of a known transmitter to attempt to re-establish communication. Potentially, each type of failure could have a number of recovery procedures specified that are attempted until one succeeds or they all fail. For many types of failures, however, the only sensible procedure may be to abort operation completely. As with recovery procedures, each type of failure may result in a different set of abort procedures being attempted in order, until one succeeds. In particular, the following four routines are applicable in all cases, and could be attempted in the stated order to minimise the cost of the failure.

Return Home In the first instance, the UAV should assess whether it has sufficient resources to return to its designated home location for convenient recovery, and do so if possible.

Land at Alternative Safe Landing Zone If returning home is not possible, then the UAV should attempt to land at an alternative designated Safe Landing Zone.

Use Sensors to identify Safe Landing Zone If a designated landing zone cannot be reached (due to insufficient resources) or identified (due to GPS failure) alternative sensors may be used to identify a safe landing zone. For example, this could be achieved using known landmarks or other visual cues, such as the colour of the ground.

Land Immediately by Controlled Descent If all else fails, and assuming flight controls are still operational, the UAV should attempt to land by controlled descent. The rate of descent should be chosen so that anyone beneath the UAV has sufficient time to identify and avoid it.

5 Search operations with smart UAVs

The capacity to search and explore a previously unknown environment is a fundamental task for a number for autonomous systems, and swarms of UAV platforms provide a unique way to explore an large and difficult to access terrains. However, their deployment, control

---

[8] In theory, we may wish to explore scenarios where UAVs can leave communication range to assume completely autonomous control. However, during test flights, we shall maintain a consistent link with a manned base station for safety considerations. Loss of communication for test purposes will be simulated in the application layer.

and coordination is not a trivial task. Controllers must account for numerous condition, including energy limitations and sensitivity to environmental hazards (in the form of natural obstacles or wind). When multiple UAVs are used, the complexity of coordination means that they are normally flown in a fixed formation relative to one another at a fixed altitude above the ground.

One way to mitigate these limitations is to automate the operations of the UAVs. If the UAVs can perform in-flight collaborations and self-organise, they offer a possibility to optimise their strategies to sense the environment in the more efficient manner possible.

When multiple UAVs are deployed, the sensory data they collect can be shared and fused to generate a complete picture of the environment — which can in turn guide the search process. This task is all the more challenging as any solution that will be proposed needs to account for limitations in terms of processing, memory storage, energy consumption, network availability and so on. Several control strategies can be considered; biology-inspired search techniques such as ant algorithms are popular due to their simplicity and low complexity, they however lack tight convergence bounds. Potential-based approaches, popular in path planning, can be extended to search and rescue operations.

Some previous work has been carried out on stochastic optimisation and control for UAVs using Markov Decision Processes (MDPs). These are stochastic since agents (UAVs) are uncertain of the external environment and its future states. The MDP maps states to actions (the policies). Optimal policies may be learned using dynamic programming solutions although more recent work has focussed on scalable, approximate, sub-optimal solutions [SC02]. The agents may also be uncertain of their own actions and those of other agents e.g. we might only exchange information between agents occasionally in order to reduce bandwidth consumption. This leads to Partially Observable Markov Decision Processes (POMDP) [LK07] where again there are challenges in providing scalable solutions. In such situations we can estimate unknown states from the available knowledge (from sensors etc). Where we have multiple heterogeneous agents, each agent may have their own POMDP and exchange knowledge from time to time [DC07]. Decentralised approaches are well suited to coordinated control, for example, token-based team coordination using POMDP has been proposed by [XSSL06], where the MDP is approximated by a POMDP. A multi-thread architecture has been employed for autonomous rotorcrafts operating under centralised control and carrying out search and rescue missions [TKF06]. Here the threads represent different optimisation goals and stochastic control was achieved via MDPs and POMDPs.

POMDPs have been used for high level decision making for UAVs tasked with search and strike [STA^{+}03], with a limited fuel supply. Performance prediction of an unmanned airborne vehicle multi-agent system has been discussed by [LD06] where UAV control agents in a dynamic multi-agent system have a set of goals such as final destination and intermediate positions. An important aspect of the UAV control problem is ad-hoc communications network management. In relation to this goal, multiagent POMDPs have been used for network routing [RG03] where the state space is: location of nodes, buffer states, and transmission success/failure status, while the actions are: send packet or remain idle. [BBH08] have developed a MDP-based solution for UAV mission control that incorporates fuel consumption into the objective function for UAV surveillance missions. In addition, the multiple goals can be conflicting and may change over time. Such issues are discussed for

UAV control by [RFF09] who adapt a MDP planner to a UAV coordination problem where plans are generated separately over time for each agent and then coordinated. However, many of these solutions are developed under the assumptions that the speed and alttitude of the UAV are constant. The quadrotors we are using, however, are low speed, low altitude and highly agile. The impact of these constraints, coupled with the tradeoff off in the resolution of the search algorithms, means that new theoretical analysis of search algorithms must be conducted [WST10].

In the scenario we envisage in SUAAVE, the UAVs require intelligent communication, command and control [FDAB05], in particular: radio network intelligence, including multi-hops, mission level task control e.g. optimal search strategies and, intelligent flight control. Control is complex in UAV environments because of unpredictable delays and losses in wireless communication alongside a need to optimise over multiple objectives and multiple parameters. Much of control theory has been developed for tightly coupled control systems in which delays are small and predictable and there is no loss; however, investigations into mechanisms by which account can be taken of the distributed and stochastic nature of systems such as we envisage are underway [DC07]. In our situation, the only control loops that are critically dependent on delay are those that involve UAV safety. For these, rather than attempt to solve the general problem, we will seek solutions that are approximate

However, while previous work has attempted to use approaches such as MDPs, POMDPs and, more generally, Reinforcement Learning, it has often focussed on one aspect of the overall optimisation problem while we attempt in SUAAVE to develop general overall search and rescue strategies. Previous work has also been mainly tested in software, via simulation, while our approach has the advantage of using a real test-bed. We therefore must develop solutions which are lightweight, efficient and safe. In our approach, the UAV control mechanisms will therefore be incorporated into a multi-agent setting, where the agents are tasked with addressing multiple heterogeneous (and possibly conflicting) goals e.g. maximising the probability of a successful search while minimising energy consumption. Types of agents we must consider include: flight (particularly separation and navigation), communication, multi-hop, safety and search agents.

Overall, search and rescue operations with a swarm of UAVs present a great number of challenges as efficiency needs to be balanced with practicality. One of our the main challenges therefore involves implementing search techniques capable of leveraging the swarm to collaborate in order to achieve a common task, while at the same time respecting the real and hard limitations of the platforms.

6 Data Fusion/Image Processing

Given payload limitations our experimental vehicles' main video sensor will be a lightweight video camera, possibly augmented with a crude (but low weight) infra-red camera and common gimbal mount. (So in particular we do not expect to carry zoom lenses.) The on-board processor is easily capable of taking a video stream at 10Hz and applying simple filters at frame rate; anything more intelligent has to be applied to the filter output. In our earlier work we have applied the Mean-Shift filter to track 'blobs' [HC07]; this worked well for tracking, but we need something more intelligent for searching. (In our tracking work the initial target was selected by a human.) We have so far experimented with the Viola/Jones algorithm for feature detection [VJ04], which is learning based; one presents a number of examples of camera shots to the learning algorithm with examples of what is to to be learnt (and without), and the system produces a small amount of code that identifies that feature in general. The Viola/Jones algorithm works well, but only when given many (say, thousands) of training examples first which have usually to be tagged by hand. It also requires a lot of computer time in this phase — so much so that only a super-computer cluster is practical [FdHC09] — but the resultant classification code can easily be run in real-time. Quite apart from the human time required then, obtaining enough footage from aerial cameras can be problematical; however the learning phase can be primed with computer-generated synthetic images to greatly reduce the amount of real footage needed. We have begun investigating the utilty of SURF descriptors [SWJT10] and we will shortly be testing related techniques, such as the use of HOG descriptors [DT05], which require a less arduous training phase.

Learning techniques are good at finding patterns, but less effective at generalisation. So, for example, a classifier designed to spot a person standing may have trouble when presneted with a picture of a person sheltering behind a wall, even if the person is in full view. Dealing with such situations is likely to require the on-board programs to search within the image, which can be time-consuming. Whether our vehicles will be able to deal with complex visual situations is still an unknown; part of our results will be to quantify how much processing is required.

Once the ‘interesting features’ have been identified in an image sequence we then have to apply some probabilistic reasoning to them, partially so we can avoid flagging up false positives and partially to sensibly combine readings taken at different times and from different positions. Dealing with continuous hypotheses has traditionally been the province of Gaussian techniques such as the Kalman filter and its many variants; non-Gaussian distributions can also be tackled in a methodical setting, but the memory and processing required to estimate an arbitrary distribution can be an issue. Whatever distributions are used, one research issue is to combine sequences of hypotheses in a reliable and effective manner. Here part of the problem is to deal with the large number of hypotheses that are typically generated with real image sequences without them swamping the processor; but another, more insiduous, problem is to combine the probability distributions without over-boosting the results [JU01]. This can occur when readings reinforce each other when they should not; most reasoning systems assume large degrees of indepence between the random variables, which may be true initially but after subsequent chains of reasoning may be invalid. Deciding which chains are valid and which are not can be subtle, but important if the combination of readings is to be effective. We shall investigate the use of various suboptimal but robust distributed data fusion algorithms based on Covariance Intersection [JU01] and its generalisation [JBU06].

7 Conclusions

SUAAVE is a little unusual among projects involvings collections of UAVs in its emphasis on the networking between vehicles. Small electric vehicles have the potential to be low-cost and applied in many situations; but their small payloads and low inertia make their control a problem and their radio environment one which is fraught with potential difficulties. Whilst buying vehicles off-the-shelf for this project we were aware of how cutting-edge such devices would be; and indeed, we have had to use time in this project modifying the vehicles for our purposes. (Nevertheless the internal architecture of the vehicles has made them relatively straightforward to deal with.)

So far the activities within SUAAVE have been focussed on getting single vehicles airborne in a safe manner, and the basic camera functionality working. With the new flying season upon us the emphasis moves to multi-vehicle cooperation: first testing the radio communications to and between vehicles, and then getting the vehicles to share information during the search process. We are also capturing images from our on-board cameras for bench testing of algorithms, with the aim of then installing those on the vehicles themselves.

Considerable emphasis has been placed on the safety of the vehicles, both by use of a 'dead-mans handle' system, but also by thinking carefully about the flight states and the transitions between them. The goal for the end of the project is to demonstrate a number (say, 10) of these vehicles engaging in cooperative tasks in a reasonably realistic outdoor environment.

References

[BBH08] B Bethke, LF Bertuccelli, and JP How. Experimental demonstration of adaptive MDP-based planning with model uncertainty. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2008.

[BPDKHO09] Oleg Burdakov, Jonas Kvarnström Patrick Dohertyand Kaj Holmberg, and Per-Magnus Olsson. Positioning unmanned aerial vehicles as communication relays for surveillance tasks. In *Robotics: Science and Systems Conference*, June 2009.

[BRS04] P. Basu, J. Redi, , and V. Shurbanov. Coordinated flocking of UAVS for improved connectivity of mobile ground nodes. In *MILCOM 2004*, volume 3, pages 1628–1634, October 2004.

[CDB04] K. Chandrashekar, M.R. Dekhordi, and J.S. Baras. Providing full connectivity in large ad-hoc networks by dynamic placement of aerial platforms. In *MILCOM 2004*, pages 1429–1436, October 2004.

[DC07] I Dusparic and V Cahill. Research issues in multiple policy optimization using collaborative reinforcement learning. In *Proceedings ICSE 2007 Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2007)*, pages 20–26, 2007.

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.

[FDAB05] E. Frew, C. Dixon, B. Argrow, and T. Brown. Radio source localization by a cooperating UAV team. In *AIAA Conf.*, 2005.

[FdHC09] Helen Flynn, Julian de Hoog, and Stephen Cameron. Integrating automated object detection into mapping in usarsim. In *Robots, Games, and Research: Success stories in USARSim (IROS 2009 Workshop)*, St. Louis, October 2009.

[HC07] Heiko Helble and Stephen Cameron. OATS: The Oxford Aerial Tracking System. *Robotics and Autonomous Systems*, 55(9):661–666, September 2007.

[HSL06] Z. Han, A.L. Swindlehurst, and K.J.R. Liu. Smart deployment/movement of unmanned air vehicle to improve connectivity in manet. In *Wireless Communications and Networking Conference 2006*, pages 252–257, April 2006.

[JBU06] Simon Justin Julier, Timothy Bailey, and Jeffrey Kent Uhlmann. Using Exponential Mixture Models for Suboptimal Distributed Data Fusion. In *Proceedings of the IEEE Nonlinear Statistical Signal Processing Workshop (NSSPW06)*, pages

160–163, Cambridge, UK, 13–15 September 2006.

[JU01] Simon Justin Julier and Jeffrey Kent Uhlmann. General Decentralized Data Fusion With Covariance Intersection (CI). In D. Hall and J. Llinas, editor, *Handbook of Data Fusion*. CRC Press, Boca Raton FL, USA, 2001.

[LD06] Z Lian and A Deshmukh. Performance prediction of an unmanned airborne vehicle multi-agent system. *European Journal of Operational Research*, 172:680–695, 2006.

[LK07] Li and T. Kunz. Cooperative node localization for tactical wireless sensor networks. In *MILCOM 2007*, 2007.

[PBGY08] S. Perumal, J.S. Baras, C.J. Graff, and D.G. Yee. Aerial platform placement algorithms to satisfy connectivity, capacity and survivability constraints in wireless ad-hoc networks. In *KMILCOM 2008*, pages 1–7, November 2008.

[RFF09] E Rachelson, P Fabiani, and Garcia F. Adapting an MDP planner to time-dependency: case study on a UAV coordination problem. In *4th Workshop on Planning and Plan Execution for Real-World Systems:Principles and Practices for Planning in Execution*, Thessaloniki, Greece, September 2009.

[RG03] Bharaneedharan Rathnasabapathy and Piotr J. Gmytrasiewicz. Formalizing multi-agent POMDP's in the context of network routing. In *HICSS 2003*, 2003.

[RZ07] I. Rubin and R. Zhang. Placement of uavs as communication relays aiding mobile ad hoc wireless networks. In *MILCOM 2007*, pages 1–7, October 2007.

[SC02] Bruno Scherrer and François Charpillet. Cooperative colearning: A model-based approach for solving multi agent reinforcement problems. In *ICTAI 2002*, pages 463–468, 2002.

[STA+03] Doug Schesvold, Jingpeng Tang, Benzir Md Ahmed, Karl Altenburg, and Kendall E. Nygard. POMDP planning for high level UAV decisions: Search vs. strike. In *CAINE*, pages 145–148, 2003.

[SWJT10] Andrew Colquhoun Symington, Sonia Waharte, Simon Justin Julier, and Niki Trigoni. Probabilistic Target Detection by Camera-Equipped UAVs. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 3–9 May 2010.

[TKF06] F. Teichteil-Königbuch and P. Fabiani. Autonomous search and rescue rotorcraft mission stochastic planning with generic DBNs. In *Artificial Intelligence in Theory and Practice*, volume 217, pages 483–492. IFIP, 2006.

[VJ04] Paul Viola and Michael Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, May 2004.

[WST10] Sonia Waharte, Andrew Symington, and Niki Trigoni. Probabilistic Search with Agile UAVs. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 3–9 May 2010.

[XSSL06] Yang Xu, Paul Scerri, Katia P. Sycara, and Michael Lewis. Comparing market and token-based coordination. In *AAMAS 2006*, pages 1113–1116, 2006.