OATS: Oxford Aerial Tracking System

Heiko Helble, Stephen Cameron
Oxford University Computing Laboratory
Wolfson Building, Parks Road
Oxford OX1 3QD, UK
`heiko.helble@comlab.ox.ac.uk`

Abstract

Small robot helicopters are becoming a popular research platform due to the availability of off-the-shelf components and their suitability for useful applications. We describe the *Oxford Aerial Tracking System* (OATS) that we are commissioning which takes a commercial airframe and low-level flight controller, and adapts these for use in applications requiring the visual tracking of ground targets. This uses a camera on a two-axis gimbal feeding images into an on-board processing system, which communicates summary information to a ground station. So far we have tested the system off the aircraft using laboratory targets and canned image sequences; we have also developed algorithms to scan a target area using a simulator program. The next step is to integrate these into the airframe, and begin testing there.

1. Introduction

Miniature autonomous unmanned aerial vehicles with vertical take-off and landing (UAVs with VTOL) capabilities have become a popular tool both for research and in practical applications. Early vehicles were notoriously difficult to control, but the recent availability of off-the-shelf flight control hardware has reduced these problems considerably. This hardware consists of low-level flight controller processors, which sit on the helicopter and take in control commands and sensor information in order to allow the vehicle to be moved in stable flight with the minimum of high-level effort. At the same time helicopter kits have become easily available (and computer hardware faster and cheaper) so that a small helicopter is capable of carrying enough of a computational payload to make it possible to do interesting mission programming on-board. We are therefore commissioning a helicopter to investigate the use of on-board image processing to achieve autonomous mission objectives. Specifically, a small camera in a two-axis motorised gimbal will be used to track a ground target vehicle across rough (but level) terrain. This task is chosen as much for the scope for interesting and scalable research as to its similarity to police aerial pursuit operations: other potential applications include military; search and rescue; surveillance; inspection of pipelines and power grids; surveying; etc.

This paper is organised as follows. Section 2. gives an overview of the relevant literature on UAVs, and section 3. describes the overall system architecture. The details of the vision algorithms used are described in section 4., followed by the modelling and tracking algorithms in section 5. and the overall conclusions in section 6..

2. Related Work on UAVs

The International Aerial Robotics Competition (IARC) (`http://avdil.gtri.gatech.edu/AUVS`) challenges participants to create a flying platform able to achieve a flight controlled without human intervention as well as a specific mission. The Technische Universität Berlin has been developing an autonomously operating flying robot named *MARVIN* since 1993, and in 2000 the team won the IARC competition with their helicopter (Musial et al., 2000). Today they are part of the *COMETS* project, which is funded by the European Community. The Swiss Institute of Technology in Zürich also runs an unmanned aerial vehicles laboratory, focusing on flight control, integrated navigation, and mission planning, and using various platforms such as unmanned helicopters, fixed wing aircraft, and airships (Tanner, 2003). The Linköping University in Sweden has a UAV research group named UAVTech. It was officially formed in January 2004, but has its roots in the WITAS UAV Project (1997-2004) (Doherty et al., 2000).

The Berkeley University VTOL UAV project is named *BEAR*, which is short for *Berkeley Aerobot Team.* BEAR currently operates six fully instrumented rotorcraft-based UAVs, implemented on various helicopter platforms. Their recent research includes obstacle avoidance in urban environments (Shim et al., 2005), collision avoidance (Shim et al., 2003), and vision-based navigation (Meingast et al., 2005). The Stanford Aerospace Robotics Laboratory has a project called Hummingbird which has demonstrated autonomous take-off, hover, trajectory following, and landing. In their latest publications they have reported inverted helicopter flight via reinforcement learning (Ng et al., 2004) and the capability to learn activity-based ground models from a moving UAV helicopter platform (Lookingbill et al., 2005). The Robotic Embedded Systems Laboratory at the University of Southern California has designed, built and conducted research with four robot helicopters. Their research includes autonomous landing (Saripalli et al., 2003), aerial sensor deployment (Corke et al., 2004), 3D navigation by using optic-flow (Hrabar et al., 2005), and autonomous flight.

Carnegie Mellon University's autonomous helicopter project has been concentrating on the development of a vision-guided robot helicopter which can autonomously carry out functions applicable to search and rescue, surveillance, law enforcement, inspection, mapping, and aerial cinematography. Major publications include (Amidi et al., 1998, Bagnell and Schneider, 2001). The Massachusetts Institute of Technology has been working with the associated Draper Research Labs on an autonomous helicopter since 1995. They have derived an architecture for autonomous helicopter control that enables the vehicle to perform agile manoeuvres. (Piedmonte and Feron, 2001) describes the learning of agile manoeuvres directly from human pilots, and in 2002 they announced the first autonomous acrobatic roll with a UAV.

3. System Architecture

The UAV is a complex electromechanical system, but for the purpose of this paper we can focus on just a few key hardware components. The airframe itself is driven by servo-motors that alter the control surfaces and regulate the speed of the main petrol engine. Helicopters are notoriously difficult to control, and thus we use a commercial Automatic Flight Control System (AFCS) that simplifies the problem of achieving steady flight. Vision processing and high-level mission objectives are dealt with using a small PC/104-based computer — essentially, a Linux PC on a single printed circuit board (PCB). On the ground a standard manual helicopter remote controller is used for take-off, landing, and emergencies, together with a laptop PC for high-level mission control and for receiving images from the helicopter.

An overview of the logical structure of the system is shown in Figure 1. In hardware terms, the box labelled 'Ground Station' refers to the laptop and radio-control equipment, and the middle three boxes 'High-Level Mission Controller', 'Object Tracking' and 'Vision Processing' are implemented on the PC/104+ aboard the UAV.

In a possible mission scenario, the UAV first flies to the desired target region. The ground station operator then selects the ground target spotted in the live video by drawing a rectangle around it using a pointer. This

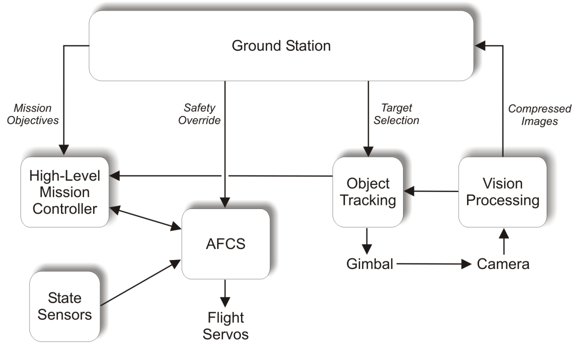
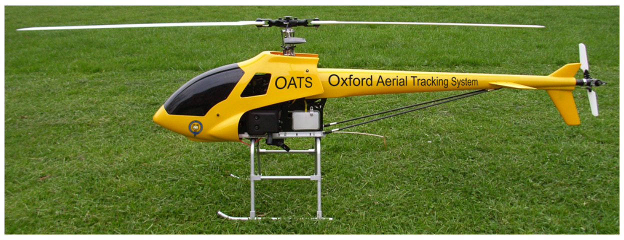
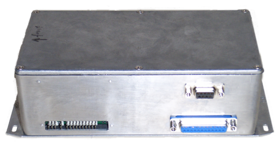


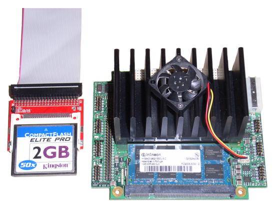
Figure 1: Logical Hardware Structure Diagram.



(a) Our JR Voyager GSR Helicopter



(b) AFCS



(c) PC/104+

Figure 2: Hardware Components.

event engages the object tracking mode, which also closes the feedback loop between vision sensor and gimbal controller so that the selected target is kept in the centre of the camera's field of view. Tracking results from the object tracker running aboard the UAV are transmitted back to the ground station in compressed form and are displayed as an overlay on top of the live video image.

3.1 Air Vehicle

The requirements of the OATS air vehicle include its ability to carry approximately 3.5kg of payload for a total duration of 20 minutes for each experiment conducted. Our robotic helicopter is based on a JR Voyager GSR (Figure 2(a)), which uses a 26cc single cylinder two-stroke engine for propulsion. It runs on a petrol/oil mixture and delivers 4.5hp, enabling the helicopter to carry payloads of up to a maximum of 6kg. With a length of 1.6m, rotor diameter of 1.8m, and a take-off weight of less than 6.5kg the helicopter is easily carried in a car to the test site. The payload compartment is lo-

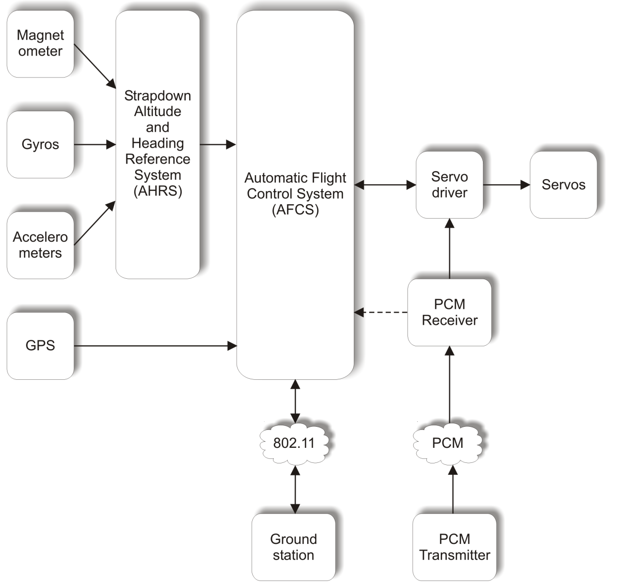


Figure 3: Automatic flight control system data flow diagram.

cated underneath the aircraft, near its centre of gravity. In manual control mode it can be flown as a regular remote controlled helicopter. For safety reasons, a switch on the transmitter can be used to regain manual control over the aircraft at all times.

3.2 Guidance, Navigation and Control

Flight control and navigation is integrated into one compact box weighing just 600g. This Automatic Flight Control System (AFCS, Figure 2(b)) is independent from the other on-board systems, communicating with those via a Fast Ethernet link (100Mbs). The AFCS stabilises the helicopter on all six degrees of freedom (x, y, z, pitch, roll, yaw) so that automatic hover flight is achieved. For state estimation, the sensory data of three inertial sensors, three gyroscopic sensors, and three magnetometer sensors are used and combined in an Altitude and Heading Reference System, as shown in Figure 3. Together with the GPS sensor signal, they are then sent to the AFCS. (All sensory data is furthermore accessible via a software interface to the other on-board systems.) The GPS data is used in two ways within the flight controller: firstly, it compensates for unavoidable sensor drift over time. Secondly, it allows for way-point navigation, so that a pre-defined flight path can be programmed prior to the start of the mission. Alternatively, new GPS coordinates can be transmitted to the UAV during flight.

3.3 Hardware Subsystems

An embedded PC/104+ processor board (Figure 2(c)) is used for on-line video processing and compression, high level mission control, and bidirectional communications relaying between the ground station and the AFCS. This board contains a Pentium-M 2GHz CPU together with 1GB of DDR-RAM and 2GB of solid-state flash memory, the latter for non-volatile storage of the Linux based operating system. The PC/104+ board features integrated Gigabit Ethernet and a built-in power supply, generating all internally required voltages from a single 5V supply line. The processor board is stacked on top of a dual slot PC-Card extension board (containing a Firewire and a WLAN adaptor card), forming a cube with an edge length of approximately 10cm and weighing around 450g.

Other hardware on the helicopter includes two separate batteries (one powering the R/C equipment; the other one powering the AFCS and PC/104+ board), an engine RPM governing circuit, a heading-hold gyroscope (used when the helicopter is operated manually), and the usual R/C equipment (PCM receiver and 5 digital high torque servos).

3.4 Aerial Imaging System

We use a consumer-grade Firewire camera which has an auto-focus objective, delivering 24 bit colour images at 30 frames-per-second (FPS) in 640×480 pixels resolution. (Our tracking experiments have so far shown that a resolution of 320×240 pixels is sufficient for the given application.) The camera is mounted on a motorised 2 degree-of-freedom gimbal, which is located at the bottom of the helicopter. The gimbal is controlled by the visual object tracker and can be rotated around its pan and tilt axis by ±60°. We send live video to the ground station both for logging and for target selection: for this we have implemented MPEG4 video compression using the freely available XviD codec. After compression, the video is wrapped into a RTSP transport stream and is then sent over the wireless network to the ground station. At a frame rate of 30 FPS, the required network bandwidth is approximately 80KByte/second.

3.5 Ground Station

The OATS UAV is controlled via a ground control station in the form of a notebook PC running the Linux operating system. Two graphical user interface (GUI) applications are run in parallel: the first displays UAV related state information and is used for navigation and guidance of the air vehicle utilising a moving GPS map display, whilst the second application controls the object tracker and displays the live video image received from the camera aboard the UAV.

3.6 Communications

Two ways for communication with the UAV are available: firstly, an IEEE 802.11b based wireless network channel operating in ad-hoc mode with 11MBit/second bandwidth, and secondly a Pulse-Code Modulation (PCM) based channel used by the R/C transmitter/receiver pair. The latter is required for take-off and landing of the helicopter, but also for security reasons in case of a system failure. Consequently, the human safety pilot can regain control over the helicopter at all times, overriding all automatic controls by flicking a switch on the transmitter which is dedicated for this purpose.

Normal 802.11b based wireless networks are limited to approximately 100m of line of sight operation. To overcome this hindrance, high gain omni-directional antennae are used in our setup on both sides. They are attached to PC-Card wireless cards (one in the ground station notebook computer, the other one in the extension board of the computing stack aboard the UAV) which are capable of transmitting signals of up to 250mW. This combination allows for range of use of up to 1000m.

4. Visual Object Tracking

Object tracking using a vision sensor from mobile robotic platforms has been an active research area for many years. However, online vision processing is a computationally demanding task and is therefore difficult to combine with the limitations inherent to airborne platforms. Advances in computer hardware and tracking algorithms have led to solutions that are powerful enough to handle this task.

The focus of our research is to develop a UAV system that can track, follow, and re-acquire arbitrary ground targets autonomously. This task requires a fast and reliable real time visual object tracker with low computational demands. A key component of the system is a geo-localisation algorithm. It calculates the target coordinates and its velocity vector in real time, allowing refinement of the target's position estimate and predicting its trajectory by analysing recorded movement patterns. For this purpose, we experiment with conventional algorithms (extended Kalman filtering) as well as novel methods of learning (Inductive Logic Programming). Another challenge not yet addressed by previous research includes target tracking in adverse environmental conditions, e.g. wind and rain. Since our experimental setup contains stationary ground obstacles, we identified prior geometric knowledge path planning as a crucial feature to be added to the system. We thereby incorporate knowledge about the flight area's geometry into the flight trajectory planner, so that fixed obstacles can be accounted for.

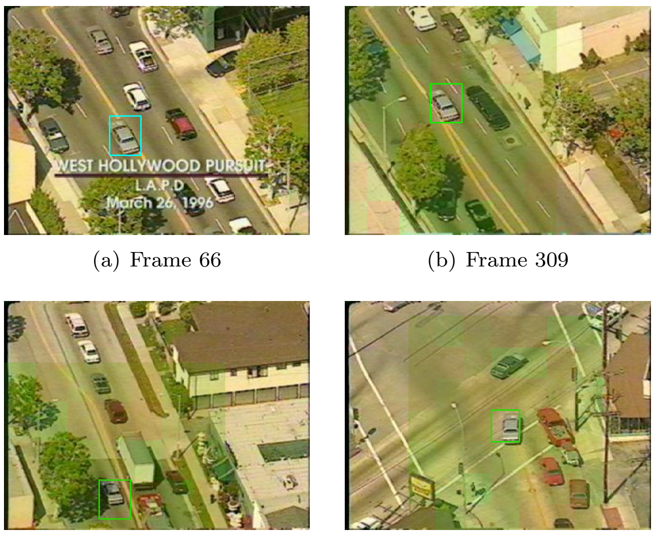
4.1 Mean-Shift Algorithm

There is a great variety of object tracking approaches found in published literature. The best known and most successful algorithms include model based tracking using geometric models, sum-of-squared differences of optical flow, Bayesian random-sampling techniques, particle and Kalman filtering, adaptive background subtraction methods, affine image registration, multiple hypothesis tracking approaches and local motion estimation. For the purpose considered in this project, we have chosen a modified version of the Mean-Shift tracking algorithm. The Mean-Shift approach was first used for visual object tracking by (Comaniciu et al., 2000). The algorithm tracks within a rectangular search window (the target) in an image by colour distribution or by texture distribution (or both). The algorithm uses an iterative gradient ascent algorithm, known as Mean-Shift, that finds a similarity peak between target and candidate positions. The correlation, or similarity, between two distributions, is expressed as a measurement derived from the Bhattacharyya coefficient. The Mean-Shift tracking algorithm endeavours to maximise the correlation between two statistical distributions. Statistical distributions can be built using any characteristic discriminating to a particular object of interest. As mentioned above, a general model might use colour, or texture, or include both. Comaniciu's implementation uses a weighted colour histogram of quantised difference values in the x and y directions. The weighting is given by the Epanechnikov kernel function (Silverman, 1986). Mean-Shift based tracking is robust to partial occlusions, clutter, rotation in depth, and changes in camera position. The algorithm takes time proportional to the product of three numbers: the search window size used (50×50 pixels here); the kernel size used (7×7); and the number of iterations until the gradient-descent operations converge (typically 3–12). The algorithm runs comfortably at 30 FPS on a 900 MHz notebook PC using a C++ implementation.

4.2 Implementation and Experiments

Modifications to the original Mean-Shift algorithm by Comaniciu include the use of gradient information (histograms of image gradient magnitude, where the image gradient is computed using standard finite difference approximations in x and y from the monochrome image), background suppression, the ability to cope with temporal target occlusion, and the capability to dynamically perform scaling operations. The described enhancements lead to measurably better performance results. A more complete description of the enhanced algorithm is given in (Bibby and Reid, 2005).

We have tested our implementation on a mixture of images taken in our laboratory and from public-domain



(a) Frame 66 (b) Frame 309

(c) Frame 833 (d) Frame 3202

Figure 4: Tracking a car in a 3351 frame long test video sequence using our extended Mean-Shift tracker.

sequences. The four images shown in Figure 4 are taken from a 3351 frame long video sequence of a moving car, filmed in bird's eye perspective from a police helicopter. The car (grey in colour) is sometimes occluded by roadside shrubs and shadows, and similar looking cars also appear at times parked along the roadside. This video sequence poses an especially tough problem for the object tracker, as long scenes of occlusion by trees and buildings occur. Furthermore, scaling operations are required due to the varying distance between camera and target. For this experiment, we configured our Mean-Shift tracker to use a dynamic search array to cope with temporal occlusions. Depending on the size of the central tracking window, multiple additional tracking windows surrounding it are used. By comparing the confidence measure of all tracking windows with one another, it is possible to identify and subsequently re-acquire the lost target as soon as it comes out of occlusion.

A Proportional-Integral-Derivative (PID) control loop is used to drive the gimbal positioning servo motors. The geometric displacement between the centre of the tracking window and the camera image centre coordinates serves as the input parameter for the PID loop. The custom built two axis gimbal construction, which is mounted below the UAV fuselage, uses two standard servo motors which require PWM signals as an input. In our setup, these signals are generated by a dedicated converter chip connected to the embedded PC's RS-232 interface port.

The target re-acquisition strategy is triggered when the visual target lock is lost for a certain period of time. The measure used to decide whether the target is being successfully tracked or not is the confidence parameter calculated by the Mean-Shift algorithm.

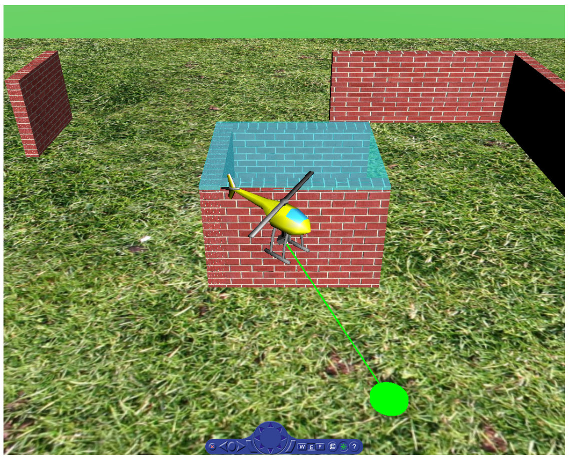


Figure 5: HeliSim virtual reality world 3-D view.

5. Modelling and Simulation

We have developed a virtual test framework named *HeliSim* that allows testing and experimenting with different terrain coverage algorithms and target search strategies. It models the UAV with its physical capabilities including its surrounding world. The helicopter and the objects around it can be viewed interactively in a three dimensional view. While the simulation is running, the state of target and helicopter can be manually manipulated (using an analogue input device) or automatically (controlling the scene objects' translational displacement and rotational angles by software). HeliSim implements a number of target search strategies which are triggered at certain events. By logging relevant data during simulation runtime, post simulation analysis can be carried out to help find more efficient search algorithms. Terrain coverage is achieved by combining helicopter translational motion with camera rotations (see §5.2).

The Mean-Shift based visual target tracker used aboard the real UAV is emulated in the simulator by a mathematical model. However, the physical limitations of the gimbal mount and the camera are exactly reproduced in the virtual world, subjecting the virtual camera to the same characteristics found on the aerial robot platform. By modelling all relevant physical properties of the existing system as accurately as possible in the simulation environment, the task of porting code from the simulator to the UAV system software becomes much more manageable.

HeliSim is implemented in Matlab and Simulink. The 3-D visualisation part (shown in Figure 5) uses the *Virtual Reality Toolbox*, resulting in a presentation-quality 3-D animation. HeliSim also features a GUI for interaction with the simulation backend. Figure 6 shows the

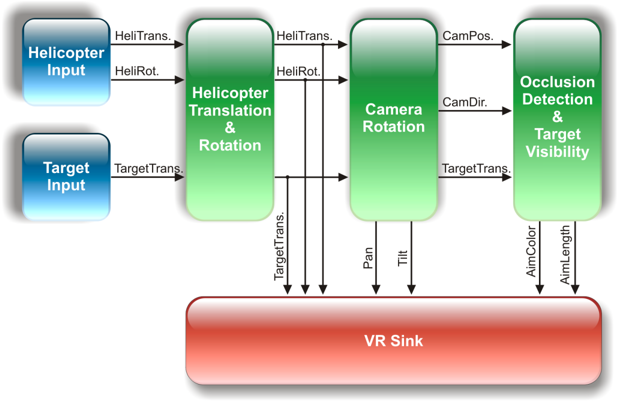


Figure 6: HeliSim global data flow diagram.

global data flow diagram between HeliSim's Simulink function blocks. The target translation vector is calculated in the *Target Input* block, and the helicopter translation and rotation vectors are determined in the *Helicopter Input* block. All three vectors are then connected to the *Helicopter Translation & Rotation* block. Each of the three modified vector output signals are then connected directly to the viewer block (*VR Sink*) as well as to the *Camera Rotation* block. The *Camera Rotation* block takes the three modified basic rotation and translation vectors and calculates the camera pan and tilt angles, which are also sent to the *VR Sink* block. Furthermore, it calculates the camera position and direction vectors and passes them together with the target translation vector to the *Occlusion Detection & Target Visibility* block. Its task is to determine whether the target is visible (i.e. located within the camera's field of view) or occluded by an obstacle. The visibility information, encoded in different colouring schemes for the virtual aim ray, is then passed to the *VR Sink* block along with a single scalar specifying the length of the aim ray.

5.1 *Target Localisation*

A sequence of six computational steps is performed in our implementation to calculate the target coordinates (see Figure 7 for details). These steps are:

1. Convert UAV 'latitude-longitude-altitude' (LLA) coordinates into 'Earth centred Earth fixed frame' (ECEF) notation.
2. Calculate local 'north-east-down' (NED) frame axis vectors.
3. Calculate UAV frame axis vectors.
4. Calculate camera frame axis vectors.
5. Calculate target ECEF coordinates by intersecting the camera frame X-axis (its optical axis) with the ground plane.
6. Convert the target ECEF coordinates into LLA coordinates.

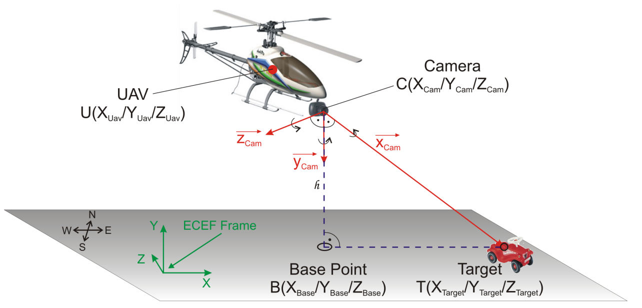

UAV
$U(X_{Uav}/Y_{Uav}/Z_{Uav})$
Camera
$C(X_{Cam}/Y_{Cam}/Z_{Cam})$
$\overrightarrow{z}_{Cam}$
$\overrightarrow{y}_{Cam}$
$\overrightarrow{x}_{Cam}$
h
N
W
E
S
ECEF Frame
Y
Z
X
Base Point
$B(X_{Base}/Y_{Base}/Z_{Base})$
Target
$T(X_{Target}/Y_{Target}/Z_{Target})$

Figure 7: Calculation of the target coordinates T. The vector along the optical axis of the camera, x_{cam}, points directly at the target. Its intersection point with ground plane E gives us the wanted target ECEF coordinates X_{Target}, Y_{Target}, and Z_{Target}.

Relating data stemming from different coordinate systems to one another requires performing a series of vector transformation operations in three dimensional space. These steps are mathematically non-trivial, but straightforward to implement.

5.2 *Terrain Coverage*

Robotic terrain coverage considers the *start-goal* problem whose solution determines a path (or trajectory) between two points so that a sensor or actuator sweeps over all points in a given region. In the context of our application this requires finding a suitable solution for the problem of target re-acquisition after the event of target loss (induced by occlusion through stationary targets). Therefore, a combination of UAV translational displacement (see §5.3) and camera rotation (see §5.4) is sought. Prominent applications that fall into this category include robotic de-mining, snow removal, lawn mowing, car-body painting, machine milling, floor cleaning, and harvesting. Ultimately, a coverage path that minimises some cost such as time is sought in all applications.

5.3 *Search by Helicopter Translation*

There are two target search algorithms implemented in HeliSim which translate the virtual helicopter: *Horizontal Search* and *Vertical Search*. They are triggered as part of a target loss strategy and can either be activated independently from each other, or in combination. Modifying the helicopter's translational displacement, various search patterns can be performed in order to retrieve a lost target.

Horizontal Search translates the helicopter in a circle around the coordinates where the target was last visible.

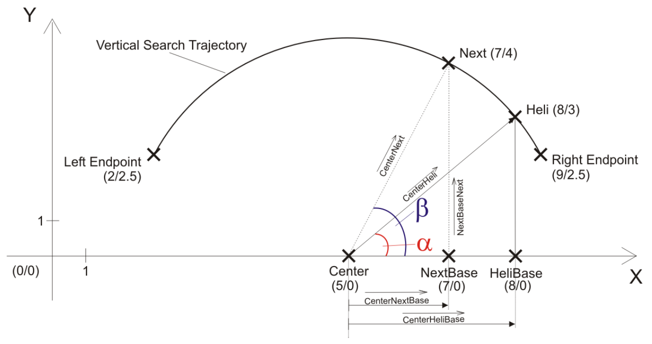


Figure 8: Calculation of the next helicopter coordinates while performing a vertical search.

This search mode only modifies the helicopter's lateral and longitudinal position so that a circular shape is described which is aligned parallel to the world frame's horizontal plane. Search parameters include circle radius, circling speed, and circling direction. Reversing the circling direction ensures that the search area covered by the camera changes from the inside region to the outside region of the circle.

The second search method, *Vertical Search*, is depicted in Figure 8. It translates the helicopter on a vertical plane that is defined by the normal vector of the world frame's horizontal plane, the coordinates of the target, and the current helicopter position in space. The translation trajectory is described by a partial circle, which has its centre at the target position and has a radius equal to the distance between helicopter and target. Its two end point coordinates are computed by the intersection between the described circle and a plane which is parallel to the world frame's horizontal plane, but vertically displaced by the altitude at which the helicopter was flying at the beginning of the search operation. Variable parameters for vertical search includes the circling speed (measured in degrees per second) and the circle radius (the distance between target and UAV). The motion direction is reversed each time one of the two end point coordinates is reached.

By combining *Horizontal Search* and *Vertical Search*, a three dimensional search sphere is described in space. The ground coordinates of the target's last valid position mark the centre of this sphere.

5.4 Search by Camera Rotation

The translational terrain search presented in §5.3 does not change the camera pan and tilt rotation angles. Thus, the camera remains in the same orientation relative to the helicopter orientation in space at the moment the target was lost by the object tracker. To improve terrain coverage of the vision sensor, pan and tilt rotations are added to the search strategy. This approach allows the search of a wider area in a shorter time. The tradeoff however is that the target might be incorrectly re-acquired by the object tracker if the camera passes too quickly over the ground area containing the target.

Two distinct camera rotation search policies are employed in HeliSim, using two different rotation axes. The *Horizontal Sweep* policy (shown in Figure 9) rotates the camera around its pan axis. When activated, the pan angle is increased in the first sweep stage in discrete steps until the maximal pan sweep angle is reached. In the second sweep stage, the pan angle is decreased until the minimal pan angle is reached. The third sweep stage finally increases the pan angle again until the original pan angle is restored. The sequence is repeated in a loop so that a continuous motion is created. Furthermore, discrete vertical angular changes can be added to the horizontal search policy, which are performed each time the horizontal rotation direction is reversed. As shown in Figure 9, the resulting overall vision sensor tool coverage is a square.

The second search policy *Vertical Sweep* is identical to the first, however the rotation axis are swapped. Accordingly, the camera is rotated around its tilt axis using optional discrete pan steps when reversing the sweep direction. Both search policies exhibit three variable parameters:

- `sweepSpeed`, describing the distance the camera aim travels on the ground per time step.
- `sweepDistance`, specifying the absolute distance of the sweep end points from the centre position on the ground.
- `perpendicularSteps`, a binary variable. If set to `TRUE`, discrete perpendicular rotation steps are added to the motion sequence when the sweep direction is reversed.

In order to control the velocity at which the camera aim ray actually travels over ground, we need to calculate the approximate distance between camera and the ground coordinates the camera is aiming at. This is achieved by using the results from the geo-localisation algorithm (determining the current ground coordinates of the target) together with the sensor data provided by the on-board GPS sensor. Therefore, `sweepSpeed` computes the gain factor of a linear function that calculates the required angular change per time step depending on the distance between camera and target. Similarly, `sweepDistance` is also determined depending on this distance. This approach ensures that the camera rotation search parameters are independent from the aircraft's height above ground and its lateral/longitudinal offset is independent from the target position.

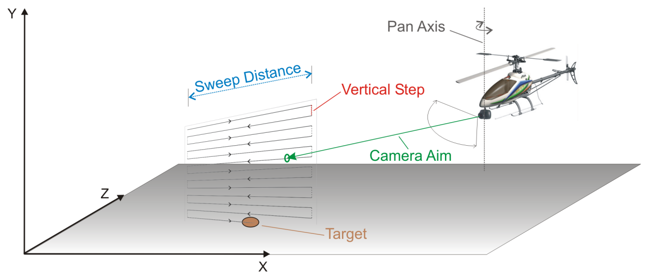


Figure 9: Horizontal target search using camera rotations.

6. Conclusions

We have presented a novel UAV research project utilising a small robotic helicopter. Although we are still commissioning the OATS helicopter itself, the components of the system can perform autonomous way-point navigation while transmitting a live video image to the ground station. Our research has focused on the development of algorithms that enable the visual tracking of intelligent targets that try to hide from the observer platform. By combining methods for geo-localisation and automatic target re-acquisition, the foundation for a useful system has been laid that could be used in applications such as law enforcement or low level surveillance tasks.

Future modifications to the existing system will include visual collision avoidance by using four low-cost cameras. Furthermore, the introduction of additional UAV's acting as a team of aerial agents in cooperative manner is being investigated.

References

Amidi, O., Kanade, T., and Miller, J. R. (1998). Vision-based autonomous helicopter research at carnegie mellon robotics institute 1991-1997. In *American Helicopter Society International Conference*, Heli, Japan.

Bagnell, J. and Schneider, J. (2001). Autonomous helicopter control using reinforcement learning policy search methods. In *International Conference on Robotics and Automation*. IEEE.

Bibby, C. and Reid, I. (2005). Visual tracking at sea. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA05)*.

Comaniciu, D., Ramesh, V., and Meer, P. (2000). Real-time tracking of non-rigid objects using mean shift. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR'00)*, volume 2, pages 142–149, Hilton Head Island, South Carolina.

Corke, P., Hrabar, S., Peterson, R., Rus, D., Saripalli, S., , and Sukhatme, G. (2004). Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, USA.

Doherty, P., Granlund, G., Kuchcinski, K., Sandewall, E., Nordberg, K., Skarman, E., and Wiklund, J. (2000). The witas unmanned aerial vehicle project. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI2000)*, Amsterdam. IOS Press.

Hrabar, S., Sukhatme, G., Corke, P., Usher, K., and Roberts, J. (2005). Combined optic-flow and stereo-based navigation of urban canyons for a uav. In *International Conference on Robotics and Automation (ICRA05)*.

Lookingbill, A., Lieb, D., Stavens, D., and Thrun, S. (2005). Learning activity-based ground models from a moving helicopter platform. In *Proceedings ICRA2005*.

Meingast, M., Geyer, C., , and Sastry, S. (2005). Vision based terrain recovery for landing unmanned aerial vehicles. In *Proceedings of the Conference on Decision and Control*.

Musial, M., Brandenburg, U. W., and Hommel, G. (2000). Marvin - flying robot for the iarc millennial event. Technical report, Technische Universität Berlin, Department of Computer Science.

Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E. (2004). Inverted autonomous helicopter flight via reinforcement learning. In *International Symposium on Experimental Robotics*.

Piedmonte, M. and Feron, E. (2001). Aggressive maneuvering of small autonomous helicopters: A human-centered approach. *The International Journal of Robotics Research*, 20(10):795–807.

Saripalli, S., Montgomery, J. F., and Sukhatme, G. S. (2003). Visually-guided landing of an unmanned aerial vehicle. In *IEEE Transactions on Robotics and Automation*, volume 19, pages 371–381.

Shim, D., Chung, H., Kim, H. J., and Sastry, S. (2005). Autonomous exploration in unknown urban environments for unmanned aerial vehicles. In *AIAA GN&C Conference*, San Francisco.

Shim, D. H., Kim, H. J., and Sastry, S. (2003). Decentralized nonlinear model predictive control of multiple flying robots in dynamic environments. In *IEEE Conference on Decision and Control*.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Chapman and Hall.

Tanner, O. (2003). *Modelling, Identification, and Control of Autonomous Helicopters*. PhD thesis, ETH. Diss. Nr. 14899.