Energy-Aware Navigation and Guidance Algorithms for Unmanned Aerial Vehicles

Ondřej Špinka[1,2]

Research and Technical Institute of Czech Air Force[1]
Department of Advanced Avionics
Mladoboleslavská 944, 197 21 Prague 9, Czech Republic
Email: spinkao@fel.cvut.cz

Zdeněk Hanzálek[2]

Czech Technical University[2]
Electrical Engineering, Department of Control Engineering
Karlovo náměstí 13, 121 35 Prague 2, Czech Republic
Email: hanzalek@fel.cvut.cz

***Abstract*—This paper presents two ways of how to tackle a problem of vehicle navigation and guidance, and focuses on evaluation of the performance and energy consumption of both methods. A simple energy-aware computing scheme, intended for control, navigation and guidance of autonomous unmanned aircraft, is proposed to try to make the most of both methods - providing good tracking accuracy whenever needed, minimizing power consumption otherwise. This scheme is based on the idea of using simple, non-demanding algorithms whenever possible and switching to sophisticated, more accurate control methods only when required by the mission profile. Simple algorithms may be executed in modules with less computing power, allowing high-performance on-board computers (needed for real-time execution of complex tasks) to be suspended, thus conserving energy. This scheme might bring noticeable benefits especially for vehicles in the micro-UAV category, where the amount of usable energy is extremely limited and the portion of energy consumed by on-board computers might be significant (up to 20%, according our measurements).**

Keywords: Unmanned Aerial Systems, Navigation, Control and Guidance Algorithms, Energy-Aware Computing, Embedded Systems

I. Introduction

Energy-aware systems and algorithms represent a rapidly growing field of interest for both academia and industry. In battery-powered embedded systems, where the amount of useful energy is strictly limited, careful husbanding of resources can bring significant benefits in terms of service life of the device. Energy-aware algorithms are used in many types of embedded systems - most prominently in the field of wireless sensor networks (focused especially on routing algorithms [1] [2] and scheduling [3]), but also in many other areas (for example [4] [5]).

In aerospace industry, energy-aware algorithms are interesting especially for small unmanned aircraft[1]. Power supplies of very small vehicles are extremely limited and the portion of energy needed for computing can be noticeable - as much as 20% of the overall power consumption (see section IV), which represents significant penalty in allowable flight time and range. Even for larger UAVs, with the mass of several kilograms, minimizing power consumption of on-board computers can be interesting. For these larger vehicles, the portion of energy consumed by on-board computers could be several percent of overall energy draw. Therefore, energy-aware control, navigation and guidance algorithms[2], designed with the goal of minimizing power consumption needed for their processing, could make noticeable difference.

Complex control, navigation and guidance methods, based on Kalman filters (such as [6] [7] [8] [9]), advanced controllers and image processing algorithms [10] [11] [12] [13] are often being used nowadays, requiring high-performance (and energy demanding) computing platforms for their real-time execution. However, there are also much less demanding, though less precise, alternatives available. Our approach is based on the fact that simple algorithms are completely sufficient for basic control, navigation and guidance tasks, while complex algorithms usually provide better accuracy, or have some other desirable properties over the simple ones. Still, there are usually many phases during a mission when great accuracy is not required. For example, in a typical reconnaissance mission profile, there are transition phases of flight between the takeoff and landing points and the objects of interest. During these phases of the mission (which may be very long), no great accuracy of control and guidance is usually required. On the other hand, during the observation phase, where the object of interest is photographed or has to be precisely tracked, more sophisticated algorithms might provide noticeable benefits. For example, very stable hover might be required while taking pictures, especially when the camera is zooming. Completely stable hovering is very hard to achieve using simple control methods, and more sophisticated algorithms using non-linear Extended Kalman Filters (EKF) or image processing might bring significant improvements.

This paper has several goals. The first one is to introduce and compare control strategies we developed and use for

[1]Generally known as Unmanned Aerial Vehicles (UAV), or Unmanned Aircraft Systems (UAS).

[2]The term "control" is usually used to describe the set of low-level control algorithms of a vehicle, such as stabilization of angular rates, attitude and velocity control, etc. By "navigation", algorithms used to determine the position, velocity and attitude of the vehicle are meant. The word "guidance" is used for algorithms used for trajectory planning and tracking, collision avoidance, etc.

Fig. 1. Manta UAV - takeoff mass 58 kg, wingspan 4 m

Fig. 2. Optoelektron I mini-UAV in flight

certain tasks in UAV control, focusing on resulting performance and power consumption. The second goal is to show experimental data on power consumption of various types of UAVs, in order to determine the portion of energy needed for computing, thus providing a rationale for introduction of an energy-aware computation scheme (section IV-A). The last goal is to propose such scheme, based on the idea of using simple algorithms whenever possible and utilizing complex computations only when needed (section IV). Section V shows some of our experimental flight data, backing our claims and results. The next section (section II) introduces our background in the area of Unmanned Aerial Systems and explains our motivation for this work.

II. Unmanned Aerial Systems

The Research and Technical Institute of the Czech Air Force (Výzkumný a Technický Ústav Letectva, VTUL [14]) has been involved in the development and production of Unmanned Aircraft Systems (UAS) since 1981. Currently, it markets a large portfolio of fully autonomous unmanned vehicles, ranging from fixed-wing aircraft to various types of rotorcraft. In the past, VTUL focused mainly on larger vehicles (Figure 1), capable of carrying large payloads (up to 30 kg) within a long range (up to 200 km). For this category of vehicles the power consumption of on-board computers was not an issue, since it was negligible compared to overall power draw.

However, current customer demand tend to prefer much smaller, lighter and cheaper vehicles, used mainly for shorter reconnaissance missions. High mobility and short turnaround time between flights are paradigm. This trend led to development of much smaller vehicles with electric propulsion, such as our Optoelektron I (Figure 2). This vehicle has a takeoff mass of 5.5 kg and its overall power consumption in cruising mode is around 125 W, out of which almost 10 W is needed for its on-board avionics. Even smaller vehicles are planned for the future, and our estimates and experiments show that the percentage of power consumption needed for on-board computers and be as high as 20% or more in the near future.

The rotorcraft are also becoming more and more popular among UAV customers, mainly because of their ability to fly in restricted areas, vertical takeoffs and landings, and the possibility to hover at a desired spot. In 2003, this led to development of a research Rotorcraft-UAV (R-UAV), called RAMA [15] [16], at the Czech Technical University (CTU) in Prague. RAMA (Remotely operated Aerial Model Autopilot) is a purely research project, never intended for commercial use. As an academical project, it is completely open-source, in the meaning that we publicly share all technical information and data at our website [17]. RAMA served as our test bed for most of the experiments presented in this paper.

III. Control, Navigation and Guidance Algorithms

A. Control Scheme

Our control scheme [15] [17] is basically the same for all types of vehicle (rotorcraft and fixed-wing aircraft) - only the internal control laws are somewhat different[3]. We decided to decompose the complex 3D control problem into independent subproblems and to solve them separately. This resulted in the hierarchical control scheme, depicted in Figure 3. Right at the start we decided to completely separate the altitude control from the rest, essentially decomposing the 3D problem into separate 1D and 2D subproblems. Hence there is an independent Altitude Control Layer (ALCL), controlling the altitude of the vehicle, and separate hierarchical structure of nested control loops, taking care of (2D) trajectory tracking (in terms of geographical position, i.e. latitude and longitude). Since both layers could be synchronized in time, this solution does not limit the ability of the system to track a complex 3D trajectory in any way.

The remaining 2D control scheme is hierarchically structured, consisting of five separate layers. The first layer (seen

[3]The main difference being the necessity to maintain a certain forward speed in case of a fixed-wing aircraft. This speed must be greater than the stall speed of the vehicle, which significantly constraints its maneuverability. A rotorcraft can hover and has much broader flight envelope.

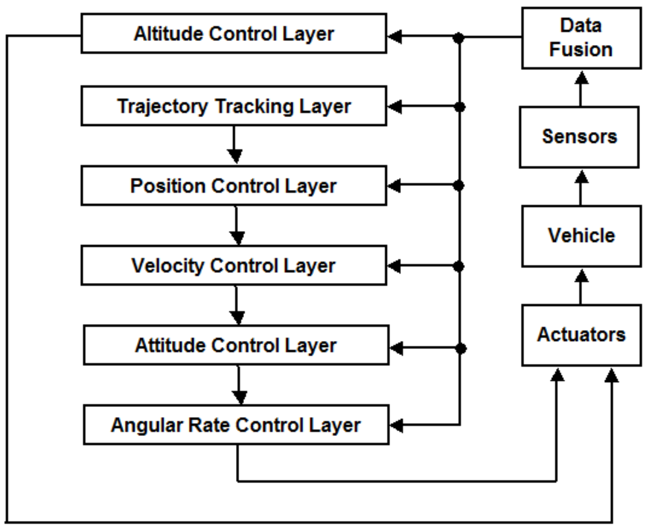

Fig. 3. Hierarchical control scheme

from the bottom to the top), the Angular Rate Control Layer (ARCL), is responsible for damping the angular rates of the vehicle (in all three axes). The second layer (Attitude Control Layer, ACL) serves for the attitude stabilization, while the third (Velocity Control Layer, VCL) controls the horizontal velocities of the vehicle. The fourth layer (Position Control Layer, PCL) is used to to stabilize the vehicle position in space. The uppermost layer (Trajectory Tracking Layer, TTL) is responsible for the vehicle guidance.

We generally use simple PID control loops in all layers, since current hardware limitations did not allow us to use more sophisticated methods in our end-products until recently (see section IV for details). These algorithms are fairly sufficient for most cases, although it would be convenient to use more complex methods like [10] [18] [11] for better accuracy in some cases (especially for rotorcraft hovering). However, in our experience with basically all types of vehicle, well-tuned PID-based control loops give good results and it is little disputable whether it really pays off to implement complex control algorithms of some sort. The hard part is to obtain good navigation data using some sensor fusion algorithms, and this is where modern data-fusion algorithms, usually based on non-linear EKF, really excel and bring significant benefits. Precision navigation positively influences all control layers, improving control quality in every respects.

B. Navigation Algorithms

Navigation algorithms serve to obtain all data necessary for control and guidance of the vehicle. Some of the data can be measured directly - for example the angular rates or accelerations - while others, such as attitude, altitude or velocity must be computed using some data-fusion algorithms. Using sophisticated data-fusion algorithms can improve not only accuracy, but also the fault-tolerance of the system, since the complete set of sensor measurements is usually redundant, and modern data-fusion algorithms (such as Kalman filters) can usually tolerate a single or multiple sensor loss and still provide a complete data set, with some loss of accuracy.

Moreover, the position and attitude of the vehicle, computed using "traditional" sensors (such as gyroscopes, accelerometers, magnetometers and GPS) can be further augmented by computer vision. This is important especially for rotorcraft, where extremely precise position measurement is essential in order to achieve stable hover (as is shown in section V). However, image processing algorithms are known for their high demand of computing resources, or require specialized hardware (such as signal processors), which can significantly increase power consumption.

Non-linear EKF (for example [6] [7] [8]), commonly used for navigation, can be quite demanding on computing power too. They are very hard to parametrize in fixed-point data types, and usually have to be computed in double-precision floating point numbers (single-precision floats often do not suffice because of possible issues with numerical stability). Having a math co-processor, ideally one supporting vector and matrix operations, is essential, since otherwise it would not be possible to run the computations at reasonable rates, required by guidance and control tasks (usually in the range of 20-100 Hz, depending on the type of the vehicle). Thus, a relatively complex MCU with significantly increased power demands is required.

On the other hand, there are also much simpler navigation algorithms, based on analytical computations of attitude angles and relying solely on the GPS for position measurement. Their demand for resources is negligible, compared to previously mentioned approaches. Such algorithms can be parametrized in fixed-point numbers when needed, and can be handled in real-time by many modern MCUs, such as the ARM7 based AT91SAM7X, used in our autopilot (see section IV-A for details).

To compare the relative demands and merits of both approaches, we developed and tested both analytical and EKF based algorithms for attitude computation. It turned out that while we were able to parametrize the analytical algorithm in fixed-point numbers and make it run on the ARM7 core in real-time at sampling rates up to 200 Hz, we could not achieve similar results with the EKF. Unfortunately, SVD decomposition and other complex matrix operations, required for this filter, are prohibitive for less-powerful computing platforms. We also had to use double-precision floats, since the algorithm became numerically unstable when single-precision floats were used. Therefore, we had to use a platform with much better computing performance for the EKF (see section IV-A).

To compare their performance, we tested both algorithms head-to-head on our RAMA test bed, on the ground and in the flight, and achieved much better results with the EKF (see section V). This outcome encouraged us to use it not only for attitude computation, but for complete navigation in our future end-user products, which led to the development of our next-generation autopilot (see section IV-A) and, consecutively, to the idea of using this advanced-but-power-demanding algorithm only when needed, conserving energy otherwise.

Source codes of both algorithms can be downloaded from our website [17].

IV. Energy-Aware Computing Scheme

A. Rationale

In section I, we mentioned that the portion of energy, needed for computing, can be significant for very small and even mid-sized UAVs. Let us show some concrete numbers, based on our experiments.

Our autopilot is based on Atmel AT91SAM7X microcontroller (MCU) (with ARM7TDMI core). The core of the MCU is clocked at 50 MHz, which is necessary to handle all the tasks it has to do in real-time (running navigation algorithms, guidance and control tasks, data logging on an Secure Digital (SD) card, on-line telemetry and communication with a ground station, control of payload, USB and serial communication on the ground, etc.). A Real-Time Operating System (Free RTOS) is used to simplify scheduling of all these tasks. This platform is not sufficient for any complex computations, since no math co-processor is present. Therefore, only simple analytical navigation algorithms and PID-based control and guidance algorithms can be used.

To allow usage of the EKF and computer vision algorithms (planned for the future), the autopilot was extended by a "heavy computing platform". Gumstix Overo micro-computer, built around the OMAP 3503 MCU (with Cortex A8 core), was selected for that purpose. It was the smallest (with 10x50 mm footprint) and most power-efficient platform we could find on the market. The "old" part of the autopilot, using the AT91SAM7X MCU, was preserved for several reasons - it is a proven and reliable, we already have vast portions of on-board software developed for it and the control system is not critically dependent on the much more complex and yet unproven Gumstix Overo platform, since the "old" autopilot serves as a hot backup whenever the Gumstix Overo assumes control (see section IV-B for details). Linux OS is used on the Gumstix Overo platform.

The power consumption of the whole autopilot (including all sensors, a data-logging device, wireless communication device with a range of 20 km, separate wireless video transmitter, etc. - but excluding actuators) is about 10 W. Out of these, approx. 5 W are consumed by the Gumstix Overo board.

To determine the percentage of energy needed for heavy computing, we measured the amount of energy, required by various types of vehicle for steady cruising (or hovering, in case of rotorcraft). The overall number includes the draw of all on-board systems, i.e. propulsion, actuators and control system. For the sake of simplicity, we used electrically-propelled vehicles only, allowing us to simply measure the battery voltage and current, giving us a fairly accurate number.

We discovered that a micro-rotorcraft (mass 250 g) can hover drawing approx. 25 W. A 1 kg electrically-propelled fixed-wing aircraft was able to fly in constant altitude at some 35 W. Optoelektron I, a 5.5 kg fixed-wing aircraft, can cruise at steady speed of 60 km/h, consuming approx. 125 W.

Cruising Power Consumption			
vehicle type	mass (g)	overall power (W)	computing power (%)
rotorcraft	200	25	20
fixed-wing	1000	35	14
fixed-wing	5500	125	4

TABLE I
Power consumption for different types of vehicle

Let us summarize these results into a table, showing the overall energy draw for each vehicle and the percentage of power needed for heavy computing (assumed to be 5 W). The results are shown in table I. From these numbers, it is obvious that for small vehicles, the percentage of overall power consumption, needed for computing, can be as high as 20%. Even for mid-sized UAVs, it is still around 4%, which is not that significant, but still noticeable and worth to be investigated. Translated into flight times, we can estimate that it can be gained around 4 to 9 min of additional flight time for small vehicles, and around 3 to 5 min for Optoelektron (based on our simulations).

B. Proposed Solution

As was already mentioned several times, our solution is based on the conclusion that a simple navigation and control algorithms can be used whenever possible (especially during the cruising phase), and switch to more accurate methods only when needed, i.e. when taking pictures, or simply whenever accurate trajectory tracking and precise vehicle stability is required.

As was explained is section IV-A, the "lower-level", less powerful ARM7 based control computer (denoted "Normal Control Computer" - NCC) is capable of executing all basic navigation, guidance, control and telemetry tasks in real-time, so the much more powerful "upper-level" Gumstix Overo platform (denoted "Advanced Control Computer" - ACC) can be switched off when great precision is not required.

In a typical mission profile, the ACC would be suspended from the start and would not be turned on until the end of the cruising phase. When the point of interest is reached and the observation phase of the mission starts, the ACC would be started and the control handover occurs. This is done by transferring current vehicle state variables (position, attitude and speed) to the ACC, where these parameters are used to initialize the EKF. Once the EKF is initialized, the ACC acknowledges the NCC that it assumes control, and the NCC enters a "dormant" mode - it is not switched off and is still executing its program as a "hot backup", but it stops issuing commands to the actuators.

The NCC monitors the performance of the ACC via periodic health check messages, sent by the ACC, and also by independently monitoring the critical flight parameters (attitude and velocity), which must always be within safe limits. In case of ACC malfunction, the MCC has the authority to turn it off (by cutting power, to prevent any possible interactions with the faulty ACC) and taking over control of the vehicle

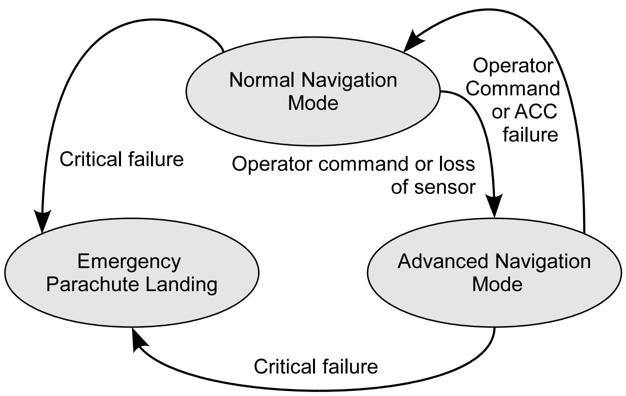

Fig. 4. Control switching and fail-safe state machine

(this is similar to the idea of the "simplex architecture", as was presented by *Seto et al* in [19]). Since the NCC is never switched off and its navigation algorithms run all the time, no synchronization of state variables is required when the control is transferred back to it. A normal handover of control, executed usually when the observation phase of the mission is finished, is done in the same way (by switching the ACC off).

Let us now show our proposed switching logic using a simple state machine (Figure 4). We assume that "normal" takeovers, aimed at conserving energy, would be initialized by the operator at the ground station (who must have the authority to decide whether extra precision is required at the moment), or could be pre-programmed in the mission profile. Emergency decisions are based upon the health-check, performed by the NCC.

The state machine (Figure 4) is self explanatory; the takeover from one platform to another occurs either when initialized by the operator, or in the case of failure. If a sensor fails when NCC is in charge, it is possible to switch to ACC and try to use the EKF to mask that failure. However, should the NCC fail, or some other "critical" failure occurs (such as the loss of all or majority of the sensors), the flight termination system[4] is activated and the vehicle lands on a parachute.

The main asset of our scheme lays, in our opinion, in its simplicity - it is very easy to implement into existing control system, and does not add any failure modes of its own, since no additional hardware is used and the take-over algorithms are extremely simple and easy to test during the hardware-in-the-loop simulations.

V. Experiments and Results

Some of our experiments, focused on determining power consumption of various types of UAVs in cruising mode, were already presented in section IV-A. In this section, we shall focus on our experiments comparing performance of

[4]All our vehicles are equipped with a flight termination system, which is totally independent on the autopilot, even having its own power source. This system can switch off the engine and fire a parachute, on which the vehicle lands safely.

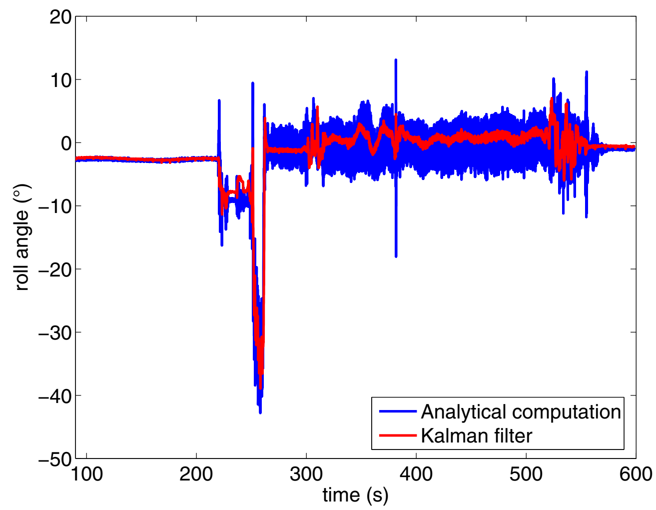

Fig. 5. Comparison of roll angle, computed analytically vs. using an EKF

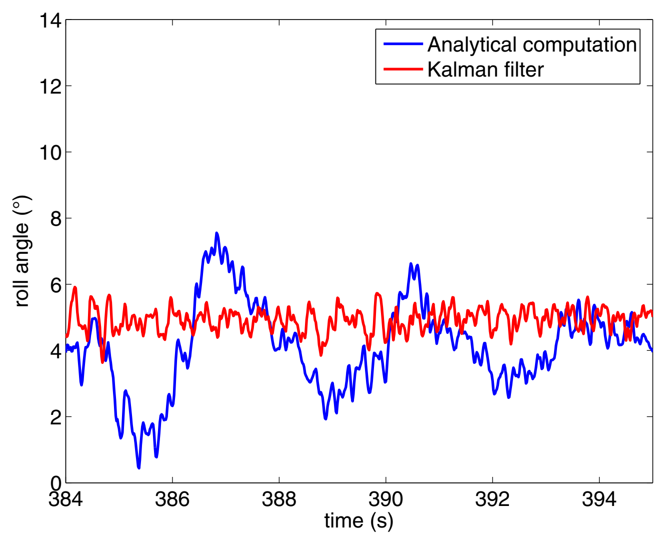

Fig. 6. Comparison of roll angle control performance

both navigation algorithms and present a simple comparison of analytical computing vs. EKF.

First, let us show a comparison of roll angle computation. Figure 5 shows a telemetry record from a test flight, with roll angle computed analytically (the blue line), and using EKF (the red line). Both algorithms were fed by the same data, which were preprocessed in the same way - the offsets of the accelerometers and gyroscopes were compensated, and all data were filtered using a fifth-order low-pass FIR filter. It can be seen that both lines match perfectly at the start and at the end of the flight, when the engine was not running. However, the in-flight data tell a different story - obviously, the EKF is able to suppress the noise in the input data much better and provides much clearer attitude measurement.

The quality of input data in turn affect the control performance, as is shown in Figure 6. The data show roll angle deviations during a rotorcraft hover. Two flights were performed with the same controller, but in one case the attitude was provided by analytical computation (the blue line) and in the second case the EKF was used (the red line). It can be seen that the attitude excursions were much smaller when EKF was running. The difference in control performance could be barely seen by naked eye during the flight; an uninformed observer, watching the vehicle from the ground, probably could not tell any difference at all. However, the data tell the whole story, and better hovering stability pays off when camera is used for photographing or filming.

So far, we were unable to properly evaluate the practical benefits of the switching logic presented in section IV-B, since our RAMA test bed is relatively large vehicle with combustion engine. Even for Optoelektron I, the benefit is rather small and hard to measure under real-life flight conditions. Smaller vehicles, for which the system was intended in the first place, are not yet ready. However, the benefits of this system are fairly easy to simulate, since for a given mission, the amount of saved energy can be easily calculated from the flight times, performed by NCC and ACC respectively. Our estimates show that for a vehicle of the 1-2 kg class, the gains in term of additional flight time could be around 4-9 minutes (approx. 6-14%), depending on the mission profile.

VI. Conclusions

The energy-aware computing scheme, presented in this paper, represents a very simple approach to conserve some energy in a demanding environment of small UAVs, where each milliwatt counts. Our aim was also to demonstrate strict relations between control algorithms and constrains of the physical system, and to share our experience with modern navigation algorithms, their merit in terms of control quality improvement and their influence on power consumption.

We believe that our work might be interesting for other researchers in the field of embedded systems, and that our idea of conserving energy by using simple algorithms whenever possible, thus allowing to suspend power-thirsty computing platforms, can be used in many other applications.

For our colleagues, interested particularly in micro and mid-sized UAVs, we hope that our comparison of simple analytical data-fusion algorithm and complex EKF, used for navigation, can be interesting - even more so, since both our algorithms are open-source and can be downloaded from our website [17]. We demonstrated practical merit of EKF, as well as its implications on computing power, resulting in increased power consumption. We also put this energy demand in perspective of overall power consumption of typical small and mid-sized UAVs.

Our future work will focus on flight-testing of proposed switching logic in micro-UAV vehicles and further evaluation of its practical benefits, which were not conclusively determined until now.

Acknowledgments

This work was supported by the Ministry of Education of the Czech Republic under project 1M0567.

References

[1] M. Younis, M. Youssef, and K. Arisha, "Energy-aware routing in cluster-based sensor networks," in *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, Fort Worth, Texas, USA, 2002, pp. 129–136.

[2] L. Longbi, N. B. Shroff, and R. Srikant, "Asymptotically Optimal Energy-Aware Routing for Multihop Wireless Networks With Renewable Energy Sources," *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, pp. 1021–1034, 2007.

[3] R. Mishra, N. Rastogi, D. Zhu, D. Mosse, and R. Melhem, "Energy aware scheduling for distributed real-time systems," in *Proceedings of the International Parallel and Distributed Processing Symposium*, Nice, France, 2003.

[4] C. D. Patel, "A Vision of Energy Aware Computing from Chips to Data Centers," in *Proceedings of the International Symposium on Micro-Mechanical Engineering*, Tsuchiura and Tsukuba, Japan, 2003.

[5] K. C. Barr and K. Asanovic, "Energy-aware lossless data compression," *ACM Transactions on Computer Systems (TOCS)*, vol. 24, no. 3, pp. 250–291, 2006.

[6] N. S. Kumar and T. Jann, "Estimation of attitudes from a low-cost miniaturized inertial platform using Kalman Filter-based sensor fusion algorithm," *Sadhana Journal*, vol. 29, no. 2, 2004.

[7] J. H. Kim, S. Sukkarieh, and S. Wishart, "Real-Time Navigation, Guidance, and Control of a UAV Using Low-Cost Sensors," *Springer Tracts in Advanced Robotics*, vol. 24, no. 7, pp. 299–309, 2006.

[8] J. Wendel, O. Meister, C. Schlaile, and G. F. Trommer, "An Integrated GPS/MEMS-IMU Navigation System for an Autonomous Helicopter," *Aerospace Science and Technology*, vol. 10, no. 6, pp. 527–533, 2006.

[9] M. Peca, "Ultrasonic Localization of Mobile Robot Using Active Beacons and Code Correlation," vol. 82, pp. 63–70, 2009.

[10] A. Bogdanov, E. Wan, and G. Harvey, "SDRE flight control for X-Cell and R-Max autonomous helicopters," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, December 2004.

[11] V. Gavrilets, B. Mettler, and E. Feron, "Nonlinear dynamic model of a small-size acrobatic helicopter," in *Proc. of the AIAA Conf. On Guidance, Navigation and Control, Montreal, Canada*, 2001.

[12] D. H. Shim, H. J. Kim, and S. Sastry, "A Flight Control System for Aerial Robots: Algorithms and Experiments," *IFAC Control Engineering Practice*, 2003.

[13] G. Buskey, J. M. Roberts, and G. Wyeth, "Autonomous helicopter hover using an artificial neural network," in *IEEE International Conference on Robotics and Automation*, May 2001, pp. 1635–1640.

[14] "The Research and Technical Institute of the Czech Air Force website," http://www.vtul.cz.

[15] O. Špinka, O. Holub, and Z. Hanzálek, "Low-Cost Reconfigurable Control System for Small UAVs," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 3, pp. 880–889.

[16] O. Špinka, J. Åkesson, Z. Hanzálek, and K. E. Årzén, "Open Physical Models in Control Engineering Education," *International Journal of Electrical Engineering Education, Manchester University Press*, vol. 47, no. 4, 2011.

[17] O. Špinka, "RAMA, an open-source UAV Autopilot - Project website," http://rtime.felk.cvut.cz/helicopter, 2007.

[18] J. Chapuis, C. Eck, M. Kottmann, M. Sanvido, and O. Tanner, "Control of Helicopters," *Control of Complex Systems*, pp. 359–392, 2001.

[19] D. Seto, B. Krogh, L. Sha, and A. Chutinan, "Dynamic control system upgrade using the simplex architecture," *IEEE Control Systems*, vol. 18(4), pp. 72–80, 1998.