Real-time Evolution of an Embedded Controller for an Autonomous Helicopter

Benjamin N. Passow, Mario Gongora, Simon Coupland, Adrian A. Hopgood

Abstract— **In this paper we evolve the parameters of a proportional, integral, and derivative (PID) controller for an unstable, complex and nonlinear system. The individuals of the applied genetic algorithm (GA) are evaluated on the actual system rather than on a simulation of it. This makes implicit a formal model identification for the implementation of a simulator. This also calls for the GA to be approached in an unusual way, where we need to consider new aspects not normally present in the usual situations using an unnaturally consistent simulator for fitness evaluation. Although elitism is used in the GAs, no monotonic increase in fitness is exhibited by the algorithm. Instead, we show that the GA's individuals converge towards more robust solutions.**

I. Introduction

Controller design and parameter identification and tuning are complex tasks where much research is being carried out [1], [2]; artificial intelligence methods, modern heuristic approaches, and even various strategies for hand design and tuning are reported in the literature.

Evolutionary computing (EC), and genetic algorithms (GA's) as a part of EC, are often used in optimisation and search problems [3], [4]. This robust and flexible method can handle complex problem domains as well as noise and can be used for multi objective optimisation [1].

An example of a control problem for a highly complex and unstable system, nonlinear, and very sensitive to external disturbances [5] is a helicopter. Generally, six degrees of freedom (DOF) are controlled by four inputs where constant control feedback from the pilot is imperative. Because of these characteristics, a controller for an autonomous helicopter must be fast in computing the control response. Active control is traditionally implemented using a combination of proportional, integral, and derivative (PID) control methods [6] which are suitable for efficient control of a helicopter [7], [8], [9].

In this paper we present our research work in evolving the parameters of a PID controller for a complex system, tested on an autonomous helicopter. The individuals of the GA are evaluated on the system itself rather than on a simulation of it. This makes implicit any system identification for the implementation of a simulator. The GA's behaviour is then analysed together with the results gathered.

Benjamin N. Passow is with the Institute of Creative Technologies (IOCT), De Montfort University, Leicester, UK, (email: benpassow@dmu.ac.uk). Mario Gongora and Simon Coupland are with the Centre for Computational Intelligence (CCI), De Montfort University, Leicester, UK, (emails: mgongora@dmu.ac.uk, simonc@dmu.ac.uk). Adrian A. Hopgood is a Professor and Dean of the Faculty of Computing Sciences & Engineering at De Montfort University, Leicester, UK, (email: aah@dmu.ac.uk).

The remainder of this paper is structured as follows. Section 2 discusses the related background to this work including some previous work in this area. Section 3 presents the system control architecture of the helicopter's embedded system and the host system the GA is running on. Section 4 introduces the GA and the experimental setup. Section 5 shows the results and their analysis and section 6 presents the conclusions.

II. Background

There is a great deal of information available on a variety of control methods, search and optimisation algorithms, and on- and off-line tuning techniques [1], [2], [10]. This section gives an overview of existing work related to the research described in this paper.

Fleming and Purshouse present in [1] a survey of EC in control systems engineering. A wide spectrum of control related applications are presented including a section on parameter optimisation and on-line applications. It is discussed that few real-time applications use EC methods for control. Additionally, it is mentioned that little work shows actual results rather than simulated results. A simulator of the corresponding system is very often used in order to evaluate the individuals' fitness within a GA.

A. Evaluation in Simulation

Sekaj and Sramek present methods based on GAs for the design of robust controllers [11]. The methods are applied to a nonlinear differential equation and compared to other methods, all in simulation. The results are promising, but no application other than in simulation has been presented.

In [10], Shim *et al* present a comprehensive study of control design for an autonomous helicopter. Three different control methodologies are compared and discussed: linear robust multi-variable control, nonlinear tracking control, and fuzzy logic control with evolutionary tuning. The genetic algorithm is used to identify and tune the consequent parameters of four controllers using fitness evaluated in a simulation. The controllers are designed and evaluated on an artificial model created from aerodynamics models.

Perhinschi [12] used a GA to identify the gain parameters of linear differential equations which are used to stabilise and control a helicopter's longitudinal channel. The results of four different GA strategies are compared by three criteria employing the fitness of the best individuals. The GA used a linearised model of a helicopter and the controller performance is not tested in simulation nor on a real system.

Mao shows in [13] a robust flight controller for a helicopter evolved using a GA. The H-infinity mixed sensitivity design approach is used for the development of the controller. The GA evolves the design parameters based on a mathematical model and the final results are tested only in simulation.

B. Evaluation on the actual System

Ahmad *et al* present an on-line GA-tuned PI controller system [14]. In this paper they present a system for tuning a heating system's controller, which is optimised in between control cycles. This is possible due to the slow response time required by such a system due to its high thermal inertia.

Nolle *et al* present a simulated annealing (SA) approach to parameter identification where solutions are evaluated on the actual system [15]. The approach shows promising results outperforming trained experts in terms of time needed and fitness of the results.

Phillips *et al* introduce a fuzzy logic based flight controller for a UH-H1 "Huey" helicopter [16]. A GA is used to find the parameters of the fuzzy controllers, evaluating the individuals on a formal numerical model of the helicopter. The resulting controller is tested in simulation and on the actual helicopter. The tests on the real system showed oscillations and a small problem in the design with the fuzzy logic controller where the simulation showed no problems.

III. Experimental System Setup

The system that is to be controlled for our experimental evaluation is a lightweight indoor helicopter. The system, its dynamics, constraints, and its embedded control system are explained in this section.

The system to be controlled in this work is a Twister Bell 47 small indoor model helicopter[1]. It is a coaxial rotor helicopter with twin counter-rotating rotors with 340 mm span, driven by two high performance direct current motors and two servos to control rotor blades' plane angles. The weight of the helicopter in its original state is approximately 210 grams and it can lift up to 120 grams. This helicopter has six degrees of freedom (DOF) controlled by four inputs. For more information on helicopters and their control and dynamics the reader is referred to [17].

In this work, only the heading controller is considered. Therefore we will only consider the dynamics related to the control of a dual rotor helicopter's heading in this paper. A helicopter's engine constantly drives the rotor. As every action has an opposite counter-action, driving the rotor causes the helicopter engine and body to turn in the opposite direction to the rotor, known as the torque effect. Usually single rotor helicopters have some kind of anti-torque system, such as a tail rotor, to counteract the torque effect. The two rotors of this dual coaxial rotor helicopter turn in opposite directions, creating opposite torque effects that cancel each other out. When both rotors are moving at the same speed a constant heading is maintained. If either rotor's speed is reduced the heading will change and lift will be reduced. The change in heading results from the differing levels of torque effect being produced by each rotor. If one rotor's speed is reduced, whilst the others speed is increased respectively, the heading will change whilst a constant amount of lift will be maintained.

[1]http://www.jperkinsdistribution.co.uk/detail.php?JPNO=6600035, accessed 27. Nov. 2007

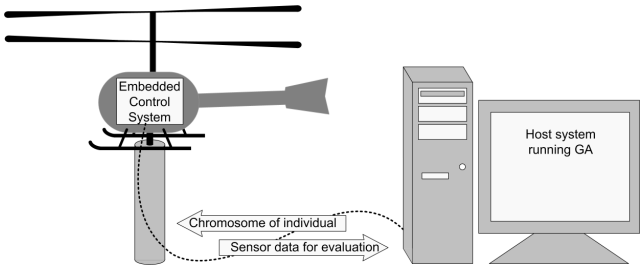


Fig. 1. Overview of experimental setup.

As part of the embedded system, a digital compass is used to determine the current heading. The sensor is connected to a microcontroller which handles all on-board computation, sensor inputs, motor outputs, and serial communication used to transfer information to and from the host computer on which the GA is running on. Figure 1 gives an overview of the system to be controlled, the host system running the GA, and the communication between them.

The control application running on the microcontroller reads all sensors, calculates all four individual PID control responses, one for each control input, and sends the overall control responses to the actuators. In this system there are 13 control cycles executed each second.

There are many control methods available that could be used for the control of this system. Classical PID control has been shown to work well controlling such a system [8]; Puntunan and Parnichkun introduce a heading direction and floating height controller for a single rotor helicopter. The control system uses a proportional plus derivative controller (PD) to maintain the heading and altitude, while a human pilot controls the horizontal movements.

In this work, the PID controller has been implemented on the embedded system using the following strategy. The proportional controller responds to an error by adjusting a control element proportional to the given error. The integral controller reacts based on the sum of recent errors, taking into account not only the amount of error but also the duration of it. The amount of previous error to take into account is limited by the integral state maximum for positive error and minimum for negative error. The derivative controller uses the increase in error from the previous error value to generate a response. The three gain values and two integral max/min parameters of the controller were adjusted until the controller performed well. Unfortunately, this task is difficult as each of the three gain values has a big influence on the other parameters.

There are a number of existing techniques for tuning the parameters of a PID controller. Methods such as simulated annealing (SA), population based incremental learning

TABLE I

HAND TUNED PARAMETERS.

Parameter	Value
Proportional gain	0.30
Integral gain	0.02
Integral state maximum	100
Integral state minimum	-100
Derivative gain	0.70

TABLE II

GA'S PARAMETER VALUE RANGE.

Parameter	Chromosome	Real Value
Proportional gain	0 - 200	0 - 2.00
Integral gain	0 - 100	0 - 1.00
Integral state maximum	0 - 16	0 - 400
Integral state minimum	0 - 16	0 - 400
Derivative gain	0 - 400	0 - 4.00

(PBIL), particle swarm optimisation (PSO) and differential evolution (DE) can be used [2]. We created an initial ad hoc hand-tuned controller to test the experimental setup. The parameters used for this controller are shown in table I. We are not going to evaluate formally the performance of the hand tuned controller against the GA at this stage.

IV. GENETIC ALGORITHM STRUCTURE

In most cases, the evaluation of the fitness of the individuals is done with a simulator. In fact, this part of the GA execution has traditionally been done in the computer where the GA is running, using either the calculations of the real problem (if it is computer based) or a simulator (if it is an external or hardware system). For this last case the main problem is to get a suitable simulator which can only be as good as the model used to create it. Getting a model then involves solving the issues of system identification and accuracy of the model vs. computational resources required to simulate it.

In the work presented here, rather than using a simulator, we use the actual system for the evaluation of the individuals for the GA. The GA itself runs on a host computer and communication between the control system and the host system is done via a serial connection. This section gives details on how the GA has been configured.

A. Solution Encoding

Every possible solution the GA might use will be encoded within a chromosome. Each individuals chromosome contains five integer values in the range specified in table II. The three gain parameters are stored in steps of one hundredth and the two integral state limits are encoded in 16 steps of 25.

B. Initial Population

The initial population of the GA could be primed using initial "good" solutions such as based on the hand tuned parameters to speed up the optimisation process. Unfortunately, we do not know if the hand tuned control parameters are near-optimal and could possibly lie in a local minima of the solution search space. Additionally we want to evaluate the performance of the GA rather than the controller, so we will initialise the system with a fresh and widespread initial population. All initial individuals are created with random chromosomes within the range specified in table II. The population size of 20 is chosen to be small enough to have a fast evaluation of each generation while providing enough individuals to maintain variety.

C. Evaluation Function

Each individual's fitness is evaluated on the real system rather than on a simulated environment. The evaluation function is shown in equation 1.

$$e = \sum (h - s)^2 \tag{1}$$

where e is the measure of error, h is the current heading and s is the setpoint. Squaring the current error in heading increases the selective pressure on the individuals causing the GA to find better solutions quicker. The sum of all the squared errors is the measure used to determine the fitness. The equivalent fitness is inverse proportional to the measure of error.

D. Selection

The selection of individuals to "survive" to the next generation is an important part of the GA. Here, the selection method is based on the roulette wheel strategy but without the possibility that an individual is chosen more than once. This method enables even the weakest individual to be chosen, although fitter individuals are more likely to be selected.

E. Genetic Operators

In this work, elitism is applied, which means that the best individual of every generation is automatically copied to the next generation without the need to be selected first. In combination with this, 20% of the old population's individuals are copied to the next generation using a roulette wheel like system.

For the crossover operator, two individuals are selected to generate one offspring. This new individual is created by taking the mean of every chromosome's loci of the parents. This method is applied in order to get 40% of the new population.

Mutation is the source of new variety. In this work, a probabilistic random mutation is used on every loci of the selected individuals to form 40% of the new population. This method of mutation uses a bell shaped probability where the chance of a small mutation is higher than the chance for a big mutation to take place.

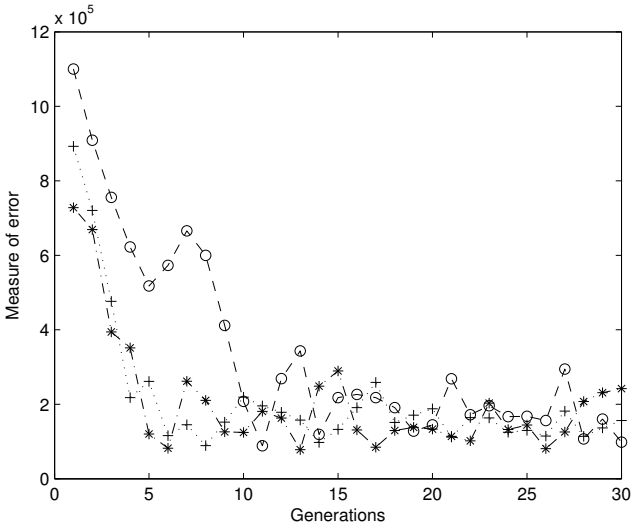


Fig. 2. Each generation's mean measure of error in three independent GA runs.

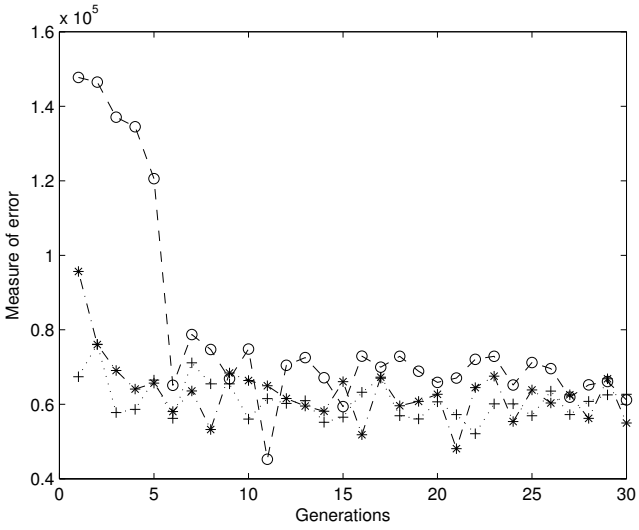


Fig. 3. Best individual's measure of error of each generation in three independent GA runs.

F. Termination Criteria

Often, there is a termination criterion in place where the GA is stopped when a certain fitness, by one or more individuals, is reached. Additionally, another termination criterion often used is where the GA is stopped when no increase in fitness is found within a defined number of generations. Because in this work we are studying the behaviour of the GA applied to a real world system at this stage we will only use a time-out as the termination criteria, stopping the GA only after an specific number of generations is reached

G. Hardware Setup

The system to be controlled, the helicopter, is attached to a ball bearing supported turntable, restricted to turn to 90° and -90° degrees from its middle position at 0°. The evaluation of one individual takes about 20 seconds and the system is cooled down in additional 20 seconds before the next individual is evaluated. Each individual is tested by perturbing the helicopter to each side and analysing the controller's reaction.

First, an individual's chromosome (the controllers' parameters) are sent to the embedded controller using a direct serial connection. Then, the helicopter starts the motors and the controller reacts on the heading error based on the parameters received. In order to test the controller's performance on a given error, the helicopter is initially perturbed by 90° to the set point by driving the two rotors with different power levels. The helicopter turns but cannot go beyond 90° as the experimental setup physically blocks it there. At this point the controller starts responding together with its actual evaluation. After 92 control cycles the evaluation and controller are paused and the helicopter is perturbed -90°, i.e. into the other direction. The controller and its evaluation are started again.

This setup, in combination with the GA running on a host computer, enables the automatic implementation and evaluation of individuals and thus the execution of the GA on the real system without any human intervention.

V. Analysis of Results

The GA was run autonomously using the real helicopter platform to measure the fitness of each individual. The results of three complete runs are presented and discussed below.

As said in the previous section, the population size was 20 individuals per generation, and the termination criteria was set to time-out after generation 30. Each individual was set to execute 189 control cycles which took approximately 20 seconds in the helicopter's embedded system, the total individual evaluation time was 40 seconds allowing the same time for cooling as for running. The time taken for the GA manipulation and scheduling parts, being run in the host computer, is negligible in comparison. Therefore, in all, from a search space of over 2 billion, the GA evaluated 600 possible individuals, taking each complete run just under 7 hours.

An initial general evaluation shows that the results are consistent with an evolutionary process as seen in Figure 2, where the mean error for each of the three complete GA runs is shown. The average error of the population drops sharply during the first third of the generations and after this the evolution process slows down as individuals converge around the best solutions in the search space.

On closer inspection of the error values, a more interesting effect is observed, which relates specifically to our approach of running the fitness function of the GA in a real system. As said in the previous section, elitism was used. With this operator the best individual of every generation is deterministically copied into the next. In usual GA experiments where a simulator is used, elitism forces a monotonic decrease in the error, and therefore increase in the fitness, of the best individual in each generation. Figure 3 shows the error of the best individuals for the three GA runs presented in this paper.

TABLE III

MINIMUM, MEAN, MAXIMUM AND STANDARD DEVIATION OF MEASURE OF ERROR OF 12 TESTS OF HAND TUNED AND BEST GA INDIVIDUALS.

	Min	Mean	Max	StD
Final best individuals:				
GA 1	64119	77243	107348	9018
GA 2	63726	77106	146877	14270
GA 3	48072	70803	92014	8925
Individuals from Figure 4:				
1.	56723	76244	112263	14241
2.	60820	77131	108455	10249
3.	61677	74230	121826	11824
4.	53257	73458	112576	10672
5.	57553	83109	180897	18010
6.	48072	70803	92014	8925

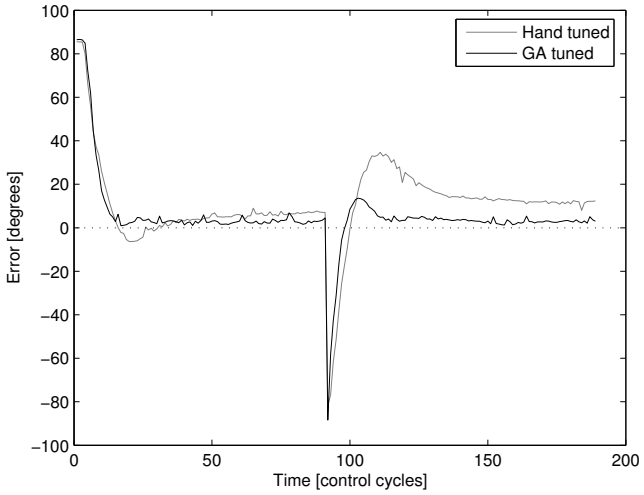


Fig. 5. GA (black) and hand tuned (gray) PID controllers response to heading perturbed by 90° at t=0 and -90° at t=92. Mean of 12 individual tests for each controller.

Although elitism was used, the error of the best individuals of every generation is not always lower or even the same as the previous one, which in theoretical systems is impossible.

In a real world system, small variations are expected when running an experiment a number of times, and that is the same for our GA fitness function. When evaluating the same individual in different generations slightly different fitness values are found. Table III shows the standard deviation for the best individuals of the three GA runs; when tested 40 times in the helicopter. This significant variability of the system can be seen graphically in Figure 4. This is the reason why Figure 3 does not show a monotonic behaviour of the error. Due to the natural uncertainties of our system, the GA cannot converge to an absolute optimal solution and therefore we have to determine what is the validity of the final solution presented by the algorithm and the validity of the fitness values.

To revisit the termination criteria in the context of fitness variability, we confirmed that at this point we cannot rely on fitness alone to stop the evolution. Not only do we have no initial idea of what a "good" or "acceptable" value of fitness is, but we have seen that reaching a fitness value in a particular generation does not necessarily mean that an "acceptable" individual has been found.

It is important to note here that due to the long time taken to run each GA, we have not tried an alternative method in which we would evaluate an individual more than once to get an average fitness rather than from a single test. To have a statistically significant average measure of fitness, relative to the variance observed, would require an extremely long time to evaluate even a single GA generation. We will see later that this is not necessary, since even with our "single test" based system we have found that the GA arrives to very suitable solutions.

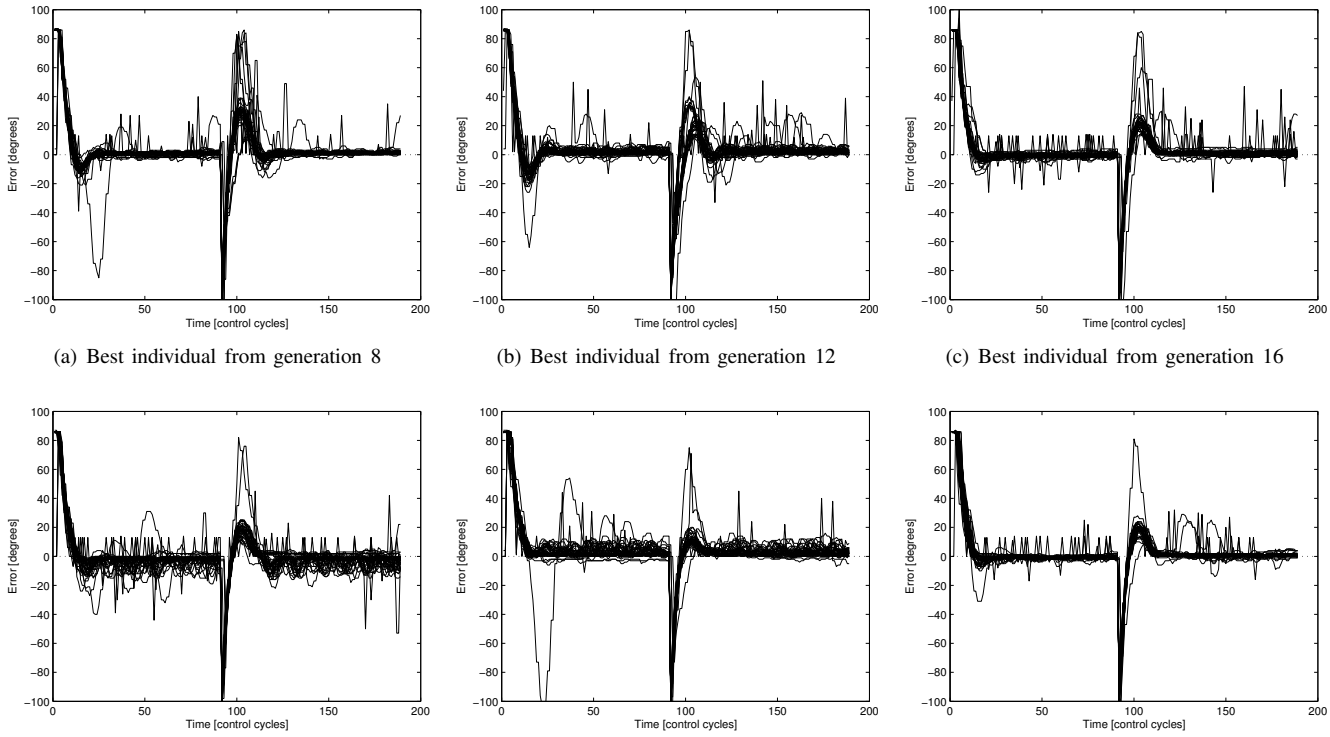
By keeping the GA running over a number of generations, even after a floor had been apparently reached in the measure of error, we have found that the GA still managed to evolve, but rather than converging to a specific "optimal" solution for an unnatural consistent simulator, it was converging toward a more "consistent" solution for a real system. This suggests that the usual concept that, when a reasonably suitable solution is found in a GA with a simulated fitness function, the algorithm can be stopped (e.g. as seen in generation 11 in Figure 3) does not apply for real system evaluated algorithms. In our case the termination criteria have to be considered from a different point of view.

In a GA based on a fitness function using the real world system, continuing the GA for a while even after apparently reaching a fitness plateau helps ensure the consistency of the final solutions. This has been confirmed by a closer analysis of the best individuals of the generations from where peaks of fitness (or low errors if seen from this point of view) occurred. Figure 4 shows the behaviour of six "best" individuals across the evolution of a GA run. Each graph shows an individual being tested 40 different times in the helicopter. Table III shows the instant fitness during evolution (in each particular generation) and the variance of the error of these individuals when tested later.

Both from the statistical analysis and a visual inspection of the graphs, it can be seen that although the particular error in the particular generation for these individuals is variable (increases or decreases with generations rather than decreasing monotonically), there is a steady increase in the consistency of the solution in terms of variability when tested multiple times on the helicopter. This suggests that the GA, although it cannot evolve towards an specific "optimal" solution, still evolves and finds its way towards a more robust solution that can perform better, which serves the exact purpose of our experiments in tuning a controller for the system.

These results and analysis show a critical and important difference between using simulated models with an unnatural consistency compared to working with a real system.

The final five fittest individuals of each of the three GA runs are shown in table IV together with their corresponding



(a) Best individual from generation 8 (b) Best individual from generation 12 (c) Best individual from generation 16

(d) Best individual from generation 21 (e) Best individual from generation 24 (f) Best individual from generation 30

Fig. 4. Control responses of six individuals of the third GA. Each graph is a composite of 40 individual tests.

TABLE IV
SOLUTIONS OF 5 BEST INDIVIDUALS OF 3 INDEPENDENT GA RUNS PLUS MEASURE OF ERROR.

PGain	IGain	IMin	IMax	DGain	Error
0.89	0.98	200	0	2.68	45237
0.83	0.43	75	0	3.13	59385
0.86	0.56	75	0	2.97	59886
0.88	0.54	125	0	3.38	61173
0.84	0.49	100	0	3.00	61901
0.93	0.34	75	0	3.67	52080
1.08	0.15	0	0	3.95	55174
1.05	0.49	25	0	3.73	56036
0.93	0.34	25	0	3.65	56045
1.15	0.58	75	0	3.95	56209
0.96	0.69	275	0	3.20	48092
0.93	0.00	0	0	3.15	51827
1.12	0.81	325	0	3.07	53210
1.02	0.56	0	0	3.55	54975
0.94	0.59	75	0	3.90	55372

measure of error. Based on these results we can identify parameters that quickly converge to a specific value region, where the parameters are more important for the controller's performance. This can be seen looking at the proportional gain. The derivative gain converged to a value region too, although not as clearly as the proportional gain parameter did. The integral gain together with the integral state maximum on the other hand seem not to have a strong influence on the performance of the controller. The integral state minimum, however, converged to zero. From the three GAs' final 60 individuals, only five did not have an integral state minimum of zero. This shows that the system is not symmetrical in nature. The helicopter controller needs a higher control response in one direction than in the other and the GA identified this.

Figure 5 shows the PID controllers' response for the hand tuned parameters and the GA's best individual's parameters, each the mean of 12 independent tests. After the first and positive perturbation, the GA optimised controller reaches the setpoint and maintains it. The hand tuned controller overshoots slightly and then maintains the setpoint with less accuracy. After the second and negative perturbation, the hand tuned controller overshoots and then very slowly approaches the setpoint. The GA based controller overshoots the setpoint too, although not as far, and then reaches the setpoint and keeps at it.

VI. CONCLUSIONS

In this work we evolved the parameters of a PID controller for a complex, nonlinear, and unstable system. The individuals of the GA were evaluated on the actual system rather than on a simulation of it. This made implicit any system identification and the implementation of a simulator. The GA's behaviour has been analysed together with results gathered.

The GA found suitable solutions as shown by the tests and an informal comparison with the hand-tuned example.

We presented the results of three independent GA runs, each evaluating 600 individuals on the actual system. We found that the GA's behaviour differs from the behaviour often seen when evaluating individuals in a simulation. Although elitism was used in the GAs, no monotonic increase in fitness is exhibited by the algorithm. Instead, we have shown that the GA's individuals converge towards more robust solutions. The cause for this behaviour is uncertainties and noise within the system. High variations in fitness when re-evaluating individuals supports this point.

The rather high variance in fitness when evaluating individuals on the real system brings additional problems. The termination criteria for a GA, where the individuals are evaluated on the real system where noise and uncertainties are present, need to be studied further.

Looking at the GA's final individuals as well as information from the execution of the GA does provide a source of information about the system. One example of this was that the GA correctly identified the asymmetrical nature of the heading behaviour, and this was represented in the parameters evolved. We showed that the behaviour in which the parameters converge may tell us even more about the system. This can be further extended to formally identify a model of a given system.

Future Work

We will continue to investigate methods for testing each individual so that we can get an average fitness and decrease the variance. A study of multiple evaluations and the strategy described here, where the GA is left running to find more robust solutions, will be considered.

Furthermore, we are going to investigate the possibility of identifying the system formally using the data collected from the GA runs, to create an accurate model of it. Based on this formal model, we can implement a simulator and be able to compare formally the GA's behaviour in simulation and on the real system.

Acknowledgment

The authors would like to thank Prof. Andrew Hugill, Director of the Institute of Creative Technologies, De Montfort University, for his support of this research project.

References

[1] P. Fleming and R. Purshouse, "Evolutionary algorithms in control systems engineering: a survey," *Control Engineering Practice*, vol. 10, no. 11, pp. 1223–1241, 2002.

[2] P. De Moura Oliveira, "Modern heuristics review for pid control systems optimization: A teaching experiment," 2005, pp. 828–833.

[3] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. University of Michigan Press, 1975.

[4] R. Haupt and S. Haupt, *Practical Genetic Algorithms*. Wiley-Interscience, 2004.

[5] I.Rojas, H. Pomares, C. Puntonet, F.Rojas, M.Rodriguez, and O. Valenzuela, "On-line adaptive fuzzy controller: Application to helicopter stabilization of the altitude of a helicopter," in *Proc. Of the International Symposium on Computational Intelligence for Measurement Systems and Applications*, Lugano, Switzerland, 29-31 July 2003.

[6] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*. Wiley, 1996.

[7] S. Saripalli, J. Montgomery, and G. Sukhatme, "Vision-based autonomous landing of an unmanned aerial vehicle," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 3, Washington, DC, May 2002, pp. 2799–2804.

[8] S. Puntunan and M. Parnichkun, "Control of heading direction and floating height of a flying robot," in *Industrial Technology, 2002. IEEE ICIT'02. 2002 IEEE International Conference on*, vol. 2, Bangkok, Thailand, 2002, pp. 690–693.

[9] E. Sanchez, H. Becerra, and C. Velez, "Combining fuzzy and pid control for an unmanned helicopter," in *Annual Meeting of the North American Fuzzy Information Processing Society*, Unidad Guadalajara, Mexico, 2005, pp. 235–240.

[10] H. Shim, T. Koo, F. Hoffmann, and S. Sastry, "A comprehensive study of control design for an autonomous helicopter," in *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, vol. 4, Tampa, Florida, USA, December 1998, pp. 3653–3658.

[11] I. Sekaj and M. Sramek, "Robust controller design based on genetic algorithms and system simulation," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, 12-15 Dec. 2005, pp. 6881–6886.

[12] M. Perhinschi, "A modified genetic algorithm for the design of autonomous helicopter control system," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, 1997, pp. 1111–1120.

[13] J. Mao, "Robust flight controller design for helicopters based on genetic algorithm," in *Proceedings of Fifth IFAC Congress*, Barcelona, 2002.

[14] M. Ahmad, L. Zhang, and J. Readle, "Online genetic algorithm tuning of a pi controller for a heating system," in *Genetic Algorithms In Engineering Systems:Innovations And Applications, 1997. GALESIA 97. Second International Conference On (Conf. Publ. No. 446)*, 2-4 Sept. 1997, pp. 510–515.

[15] L. Nolle, A. Goodyear, A. Hopgood, P. Picton, and N. Braithwaite, "Improved simulated annealing with step width adaptation for langmuir probe tuning," *Engineering Optimization*, vol. 37, no. 5, pp. 463–477, 2005.

[16] J. Chang Doo, S. Seung Il, K. Sang Keun, C. Phillips, C. Karr, and G. Walker, "Helicopter flight control with fuzzy logic and genetic algorithms," *Engineering Applications of Artificial Intelligence*, vol. 9, no. 2, pp. 175–184, 1996.

[17] S. Coyle, *The art and science of flying helicopters*. Arnold, 1996.