Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Hardware Implementation of Aircraft Landing Controller by Evolutionary Computation and DSP

Jih-Gau Juang*, **Hou-Kai Chiou**

**Department of Communications and Guidance Engineering, National Taiwan Ocean University, Keelung, Taiwan (e-mail: jgjuang@mail.ntou.edu.tw)*

Abstract: This paper presents a hybrid control scheme for the aircraft automatic landing system. PID control law is adopted in the controller design. Disturbance adaptive capability is demonstrated through hardware in the loop simulations. The control scheme uses PID controller with evolutionary computation technique. Control gains are selected by real-valued genetic algorithms. Different crossover methods are utilized to improve the performance of conventional automatic landing system. Hardware implementation of this intelligent controller is performed by a DSP with VisSim platform. The proposed intelligent controllers can successfully expand the controllable environment in severe wind shear conditions.

1. INTRODUCTION

Aircraft landing control based on simulation has been done by many researchers (Iiguni *et al.*, 1998; Izadi *et al.*, 2003; Chaturvedi *et al.*, 2002; Ionita *et al.*, 2002; Nho *et al.*, 2000; Jorgensen *et al.*, 1991). These researches are all software simulations. Some researchers have applied intelligent concepts to the problem of intelligent landing control, but these intelligent concepts are not adaptive to various wind disturbance conditions. In the research of intelligent flight control, neuro-control is the most used technique. Juang (2002, 2003) had presented a sequential learning technique that uses conventional neural network with back-propagation through time algorithm in landing control successfully. According to Boeing's report (Boeing Publication, 2000), 67% of the accidents by primary cause are due to human factors and 5% are attributed to weather factors. By phase of flight, 47% accidents are during final approach or landing. It is therefore desirable to develop an intelligent automatic landing system (ALS) that expands the operational envelope to include safer responses under different flight conditions. In this study, robustness of the proposed controller is obtained by choosing optimal control gain parameters that allow a wide range of disturbances to the controller. Here, we propose a control technique that can overcome environment problem and make the controller more robust and adaptive to various wind disturbance conditions. In this paper, we present two control schemes for hardware simulation, a conventional PID controller and a PID controller with evolutionary computation (EC). To improve the performance of the automatic landing system, EC is implemented into the controller scheme, which can search more suitable control gains of the pitch autopilot. Furthermore, we download the code to the target DSP chip, and run it on the VisSim platform. Finally, the comparisons of results by simulation and hardware in the loop experiments are presented.

EC is the general term for several computational techniques based on the evolution of biological life in the natural world. In computer science evolutionary computation is a subfield of artificial intelligence involving combinatorial optimization problems. Evolutionary computation includes genetic algorithms (GA), evolutionary programming (EP), evolution strategies (ES), genetic programming (GP), and classifier systems (CS). Despite these subfields of evolutionary computation being developed independently over several decades, most of these techniques are similar in spirit, but differ in the details of their implementation and the nature of the particular problem to which they have been applied. The GA is search procedures based on the mechanics of natural genetics and natural selection. Compared to GA, the EP typically only uses the mutation process without the binary-coded and reproduction (or crossover) processes. The efficiency of ES also depends on the size of mutation strength. The GP is a kind of operators, which is based on the basic function of the definition of the working purpose, on the basis of limiting conditions of questions, through evolutionary computation to generate optimal procedure modeling automatically. The difference between GA and GP is that the GA is by way of coding and the GP is by way of procedure. The CS is a kind of adaptive rule-based system, which can study complicated rules from the changes of external environment and adjusts progressively to strengthen own inherent knowledge. The most widely used form of evolutionary computation is the generic algorithm.

The GA was proposed by John Holland in 1962, which is an optimization and search technique based on the principles of genetics and natural selection. By 1975, Holland mentioned the most basic principle of GA in "Adaptation in Natural and Artificial System" (Holland, 1975). In the same year, De Jong showed the usefulness of the GA for function optimization and made the first concerted effort to find optimized GA parameters. The GA generally only involves techniques of implementing mechanisms, such as reproduction, crossover, mutation, fittest function, etc. Via reproduction, crossover, and mutation steps, the GA can generate next generation to reach purpose of evolution. According to level of fitness value, the GA retains the fine one and eliminates the inferior popu-

978-3-902661-00-5/08/$20.00 © 2008 IFAC

10.3182/20080706-5-KR-1001.1173

lations. Therefore, it is widely used to solve optimal problems recently. It can search many points at the same time and is not apt to fall into local optimal solution. The GA has been used for a wide range of applications, as well as for specific applications focused on a specific requirement. So far many new improving methods focus on making GA more efficient and widely increasing parameter searching ability. Here we put focus on crossover principle of different ECs. We utilize five crossover principles (Adewuya, 1996; Michalewicz, 1992; Eshelman *et al.*, 1993) under wind disturbances to search optimal control gains, and compare the differences between these principles. Different wind disturbances are also implemented into the flight simulations. This study also utilizes the *VisSim software* and *TI C2000 Rapid Prototyper* to develop an embedded control system that uses a DSP controller. Thus, realization of on-line real-time control can be achieved.

2. SYSTEM MODEL

At the aircraft landing phase, the pilot descends from the cruise altitude to an altitude of approximately 1200 ft above the ground. The pilot then positions the aircraft so that the aircraft is on a heading towards the runway centerline. When the aircraft approaches the outer airport marker, which is about 4 nautical miles from the runway, the glide path signal is intercepted, as shown in Fig. 1. As the airplane descends along the glide path, its pitch, attitude, and speed must be controlled. The descent rate is about 10 ft/sec and the pitch angle is between -5 to +5 degrees. Finally, as the airplane descends 20 to 70 feet above the ground, the glide path control system is disengaged and a flare maneuver is executed. The vertical descent rate is decreased to 2 ft/sec so that the landing gear may be able to dissipate the energy of the impact at landing. The pitch angle of the airplane is then adjusted, between 0 to 5 degrees for most aircraft, which allows a soft touchdown on the runway surface. A simplified model of a commercial aircraft that moves only in the longitudinal and vertical plane is used in the simulations for implementation ease (Jorgensen *et al.*,1991).

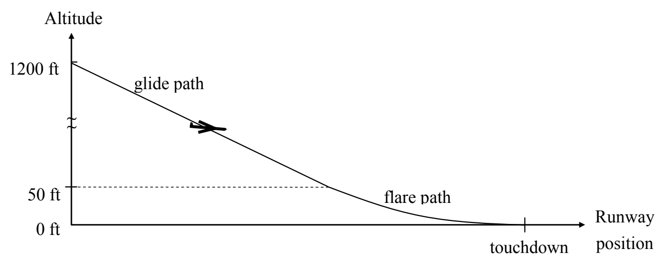

Fig. 1. Glide path and flare path

To make the ALS more intelligent, reliable wind profiles are necessary. There are four wind shear forms which are mostly used for aircraft response studies. In this study one of these forms (Berkmann *et al.*, 1995) is used for its demonstration ease. The wind shear velocity components include longitudinal (W_x) and vertical (W_h). The model is given as follow

$$W_x = k \cdot A(x)\,,$$
$$W_h = k \cdot \left(\frac{h}{h^*}\right) \cdot B(x)\,. \qquad (1)$$

The distribution of the horizontal wind versus the horizontal distance is given by

$$A(x) = \begin{cases} -p + ax^3 + bx^4 + cx^5 & x \le x_1 \\ r\left(x - \dfrac{x_3}{2}\right) & x_1 \le x \le x_2 \\ p - a(x_3 - x)^3 - b(x_3 - x)^4 - q(x_3 - x)^5 & x_2 \le x \le x_3 \\ p & x \ge x_3 \end{cases} \qquad (2)$$

and the distribution of the vertical wind versus the horizontal distance is given by

$$B(x) = \begin{cases} dx^3 + ex^4 + sx^5 & x \le x_1 \\ -51\exp\left[-c\left(x - \dfrac{x_3}{2}\right)^4\right] & x_1 \le x \le x_2 \\ d(x_3 - x)^3 + e(x_3 - x)^4 + s(x_3 - x)^5 & x_2 \le x \le x_3 \\ 0 & x \ge x_3 \end{cases} \qquad (3)$$

where x is the horizontal position (ft) relative to ground of the aircraft, x_1, x_2, and x_3 are various horizontal distances (ft) measured from the initial position, h is the vertical altitude (ft) of the aircraft, h^* is the reference altitude (ft), k is the intensity of the wind shear/downdraft combination, a, b, c, d, e, p, q, r, and s are relevant parameters of the wind shear. The parameters are shown in Table 1. Fig. 2 shows the wind shear profiles with wind speed of 30 ft/sec.

Table 1. The relevant parameters of the wind shear model

Parameter values for wind shear model	
x_1=1300 ft	x_2=3300 ft
x_3=4600 ft	h^*=1000 ft
$a \approx$ -8.55712x10^{-8} sec^{-1}ft^{-2}	$b \approx$ 1.16943x10^{-10} sec^{-1}ft^{-3}
c=0.95x10^{-12} sec^{-1} ft^{-4}	$d \approx$ 6.45597x10^{-8} sec^{-1}ft^{-2}
$e \approx$ -9.97370x10^{-11} sec^{-1}ft^{-3}	k=1
p=42 ft sec^{-1}	r=0.04 sec^{-1}
$q \approx$ -3.87834x10^{-14} sec^{-1} ft^{-4}	$s \approx$ 3.32076 x10^{-14} sec^{-1} ft^{-4}

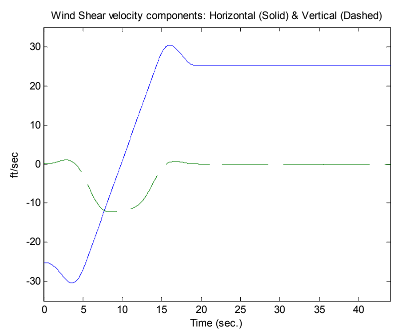

Fig. 2. Profile of wind shear model

VisSim is a Windows-based program for the modeling and simulation of complex nonlinear dynamic systems (Visual Solutions, 2002). VisSim combines an intuitive drag & drop block diagram interface with a powerful simulation engine. The visual block diagram interface offers a direct method for constructing, modifying and maintaining system models. The simulation engine provides fast and accurate solutions for linear, nonlinear, continuous time, discrete time, time varying and hybrid system designs. In here, we build an aircraft dynamic model under VisSim software, and realize the conventional PID controller by the same manner. Then, the intelligent controller design using C language and its realization by DSP are presented.

Since the invention of the transistor and integrated circuit, digital signal processing functions have been implemented on many hardware platforms ranging from special-purpose architectures to general-purpose computers. It was not until all of the functionality (arithmetic, addressing, control, I/O, data storage, control storage) could be realized on a single chip that DSP could become an alternative to analog signal processing for the wide span of applications that we see today. In this study, we use TI TMS320LF2407 chip to perform our task. The 2407A devices offer the enhanced TMS320DSP architectural design of the C2xx core CPU for low-cost, low-power, and high-performance processing capabilities. Moreover, it offers suitable array of memory sizes and peripherals tailored to meet the specific performance points required by various applications. The TMS320LF2407 operates at 40 MHz (40 MIPS), has 4 to16 PWM output channels and has serial communication capabilities. In addition, the TMS320LF2407 contains a 10-bits analog-to-digital converter (ADC) having a minimum conversion time of 500 ns that offers up to 16 channels of analog input. Furthermore, the working process and externals of the eZdspTMLF2407A board are shown in Fig. 3 and Fig. 4, respectively.

There are three basics steps in the development of a DSP algorithm: 1) Create the system you want to execute on the target DSP; 2) Generate the C source code from the system; 3) Compile and link the C source code to produce an executable file. If step 1 is performed using available blocks form VisSim software, then steps 2 and 3 are automatically performed by VisSim. But most intelligent methods do not exist in VisSim toolbox. So, an intelligent controller design using C language and its realization with DSP are presented in next section. Detail description of the aircraft model and control structure can be found in (Juang *et al.*, 2003).

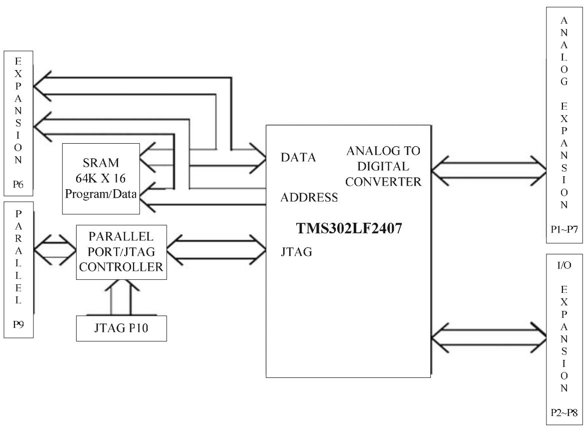

Fig. 3. Working process of the eZdspTMLF2407A board

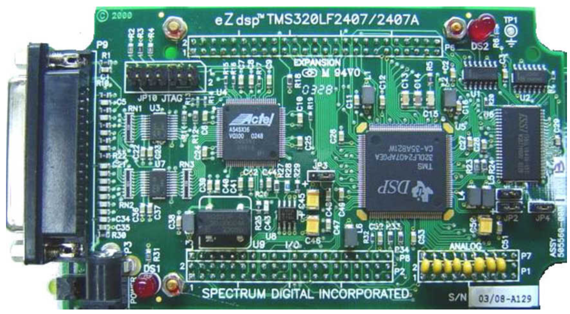

Fig. 4. Externals of the eZdspTMLF2407A board

The aircraft starts the initial states of the ALS as follows: the flight height is 500 ft, the horizontal position before touching the ground is 9240 ft, the flight angle is -3 degrees, the speed of the aircraft is 234.7 ft/sec. Successful touchdown landing conditions are defined as follows:

(a) $-3 \le \dot{h}_{TD}(T) \le -1$ (ft/sec)

(b) $-300 \le x_{TD}(T) \le 1000$ (ft)

(c) $200 \le V_{TD}(T) \le 270$ (ft/sec)

(d) $-10 \le \theta_{TD}(T) \le 5$ (degrees)

where T is the time at touchdown, $\dot{h}_{TD}$ is vertical speed of the aircraft at touchdown, x_{TD} is the horizontal position at touchdown, V_{TD} is the horizontal speed, θ_{TD} is the pitch angle at touchdown. Performance of a conventional PID controller that uses available blocks is presented in Table 2. The controller can only guide the aircraft flying through wind speed from 0 ft/sec to 15 ft/sec.

Table 2. The results of successful touchdown from using conventional PID controller in wind shear condition under VisSim Platform

Wind speed	Horizontal velocity (ft/sec)	Landing point (ft)	Aircraft vertical speed (ft/sec)
10	234.6779	916.2113	-2.9268
11	234.6779	929.8227	-2.9016
12	234.6779	943.9033	-2.8767
13	234.6779	957.9840	-2.8519
14	234.6779	972.0647	-2.8259
15	234.6779	986.1454	-2.8013

3. EVOLUTIONARY COMPUTATION

The GA's good properties do not stem from the use of bit strings. GA based on real number representation is called real-valued genetic algorithm (RGA), which would seem particularly natural when optimization problems with variables in continuous search spaces are tackled. A chromosome is a vector of floating point numbers whose size is kept the same as the length of the vector, which is a solution to the problem. The operation mechanisms, reproduction, crossover, and mutation are operated by real number also. In recent years, a lot of researchers have done much improvement for crossover and mutation, in order to expect GA to have better performance. In this study, tolerance limits of wind shear are used as the performance index. In addition to utilize GA with five kinds of crossover principles to search the parameters of the pitch autopilot, we also compare the differences between these five principles.

3.1 Adewuya Crossover Principle

We utilized roulette wheel selection to choose better parents, which is according to the fitness function of populations. For each generation, the reproduction operator chooses populations that are placed into a mating pool, which is used as the basis for creating the next generation. Then, enter the next stage, crossover. The crossover in the RGA is an important process. The first principle we used is called the Adewuya crossover, which was proposed by Adewuya (1996). The process is divided into three steps, as shown below.

Step 1：Randomly choose a gene from each individual of a matching pair in parent generation, $P_{m\alpha}$ and $P_{n\alpha}$, as crossover site.

$$pattern_1 = \left[p_{m1} \quad p_{m2} \quad ..... \quad p_{m\alpha} \quad ..... \quad p_{ms} \right] \tag{4}$$

$$pattern_2 = \left[p_{n1} \quad p_{n2} \quad ..... \quad p_{n\alpha} \quad ..... \quad p_{ns} \right] \tag{5}$$

Step 2：Calculate new values of these selected genes as follows, where β is a random number and $0 \le \beta \le 1$.

$$p_{new1} = (1-\beta) \cdot p_{m\alpha} + \beta \cdot p_{n\alpha} \tag{6}$$

$$p_{new2} = \beta \cdot p_{m\alpha} + (1-\beta) \cdot p_{n\alpha} \tag{7}$$

Step 3：Replace $P_{m\alpha}$ and $P_{n\alpha}$ with P_{new1} and P_{new2}, respectively. The genes in the right side of the crossover site exchange with each other, which will obtain new offspring.

$$Newpattern_1 = \left[p_{m1} \quad p_{m2} \quad ..... \quad p_{new1} \quad ..... \quad p_{ns} \right] \tag{8}$$

$$Newpattern_2 = \left[p_{n1} \quad p_{n2} \quad ..... \quad p_{new2} \quad ..... \quad p_{ms} \right] \tag{9}$$

Finally an important process is the mutation, which permits the introduction of extra variability into the population. We pick out a population randomly, and change their gene information, but the new offspring must be in the range established after adding gene information. We use real number mutation process as follow

$$x_{new} = x_{old} + s \cdot rand\_noise \tag{10}$$

where s is the random value between 0 to 1.

3.2 Arithmetical Crossover Principle

The second principle is the arithmetical crossover (Michalewicz, 1992). The reproduction and mutation that we used are the same as in section 3.1. The arithmetical crossover makes the mating pair pull away or get closer. The process is shown below.

$$\text{Pull-away} \quad \begin{aligned} x_1' &= x_1 + \sigma \cdot (x_1 - x_2) \\ x_2' &= x_2 - \sigma \cdot (x_1 - x_2) \end{aligned} \tag{11}$$

$$\text{Get-closer} \quad \begin{aligned} x_1' &= x_1 + \sigma \cdot (x_2 - x_1) \\ x_2' &= x_2 - \sigma \cdot (x_2 - x_1) \end{aligned} \tag{12}$$

where x_1 and x_2 are the parents, x_1' and x_2' are the new offspring, and σ is a random and positive small real value. In addition, we can also use either (11) or (12) with $-1 < \sigma < 1$, which can determine pull away or get closer by the sign of σ.

3.3 Average Crossover Principle

The third principle is the average crossover (Michalewicz, 1992). The reproduction and mutation that we used are the same as in section 3.1.The average crossover uses a simplified model with (6) and (7) where β is 1/2. It can be obtained as below.

$$p_{new} = \frac{1}{2} \cdot (p_{m\alpha} + p_{n\alpha}) \tag{13}$$

where $P_{m\alpha}$ and $P_{n\alpha}$ are the parents, P_{new} is the new offspring.

3.4 Convex Crossover Principle

The fourth principle is the convex crossover. The reproduction and mutation that we used are the same as in section 3.1. The convex crossover is shown below.

$$x_{new} = \gamma \cdot x_j + (1-\gamma) \cdot x_k \tag{14}$$

where x_j and x_k are the parents, x_{new} is the new offspring, γ is a random and small real value.

3.5 Blend Crossover Principle

The fifth principle is the blend crossover. The reproduction and mutation that we used are the same as in section 3.1. The blend crossover (BLX-α) was proposed by Eshelman and Schaffer (1993). It is a prominent crossover operator for RGA, and excels in optimization of a number of standard separable functions with multimodality. Figure 5 shows illustration of the BLX-α crossover, where I is for d_i, P_1 and P_2 are for x_i^1 and x_i^2, respectively. It generates offspring as following process.

$$X_i^1 = \min(x_i^1, x_i^2) - \alpha \cdot d_i \tag{15}$$

$$X_i^2 = \max(x_i^1, x_i^2) + \alpha \cdot d_i \tag{16}$$

$$d_i = \left| x_i^1 - x_i^2 \right| \tag{17}$$

where x^1 and x^2 are chosen randomly from the population, x_i^1 and x_i^2 are the i-th elements of x^1 and x^2, respectively. The value of each element x_i^c of the offspring vector x^c is uniformly sampled from the interval $\left[X_i^1, X_i^2 \right]$. α is a positive parameter, which is suggested to be 0.5.

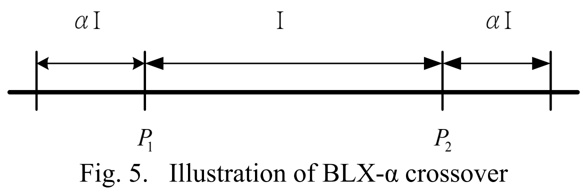

Fig. 5. Illustration of BLX-α crossover

In intelligent systematic field, the ability of function optimization makes the RGA effective for adjusting neural network weights or some parameters of other evolutionary algorithms and techniques. For this reason, robustness of the proposed controller is obtained by choosing optimal control gains that allow a wide range of disturbances to the controller in this study. Therefore, the RGA is suitable for determining the control parameters, which give aircraft better adaptive capability in severe environments. Table 3 shows the comparison of different crossover principles in wind shear conditions.

Table. 3 Comparison of different crossover principles of RGA in wind shear condition

	CPU times of GA over total time	Error of altitude command and actual altitude	Maximal wind shear intensity (ft/sec)
Adewuya crossover	1.36%	6.94%	67
Arithmetical cross-over	1.18%	7.13%	68
Average crossover	0.88%	6.52%	64
Convex crossover	1.18%	6.67%	63
Blend crossover	1.22%	7.01%	65

4. IMPLEMENTATION BY DSP CHIP

The core of VS-ECD2407 is TI TMS320LF2407, which is a 16 bits fixed-point DSP. While designing the controller of an aircraft, it must consider the problem of the fixed point. Because molds of VisSim are all floating-point operation, we must match VisSim/Fixed-Point software to design molds of fixed-point flight controller. Figure 6 shows the DSP development procedure entirely, which is divided into the following several steps.

Step 1：In VisSim major software, the fixed-point controller molds of an aircraft are designed by VisSim/TI C2000 Rapid Prototype and VisSim/Fixed Point.

Step 2：CCStudio can make fixed-point controller molds to do compiling, analysis, debugging, and demonstrating, and generate *.c code and *.out code finally.

Step 3：Generate DSP controller molds which include *.out code, and replace original fixed-point controller molds.

Step 4：Download *.out code to TI TMS320LF2407 embedded flash memory by JTAG.

Step 5：Utilize DSP controller to control automatic landing system and show real-time relevant flight behavior.

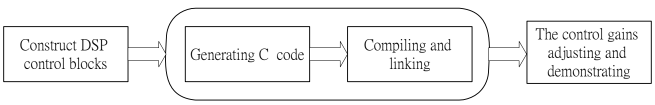

Fig. 6. The flow chart of VisSim/DSP procedure development

Through above-mentioned DSP controller's procedure of developing, whole real-time DSP hardware in-the-loop mode is shown in Figure 7. From VisSim development platform, it will transmit information, altitude, altitude rate, altitude and altitude rate commands to the VS-ECD2407 via JTAG, a connection between the computer and the VS-ECD2407. After DSP processing, it passes the pitch command and adjusts the angle of elevator back to pitch autopilot in VisSim via JTAG, and enable an aircraft to follow landing trajectory to land.

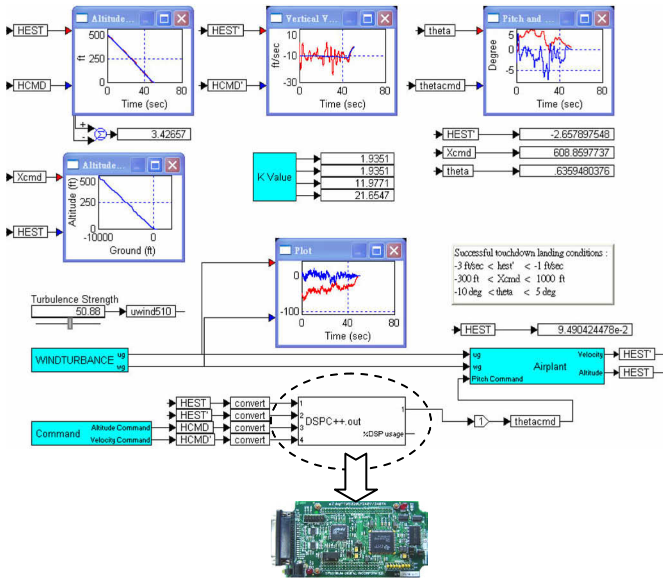

Fig. 7. Hardware in the loop

DSP has characteristics, such as fast operations, powerful instruction system, fixed addressing ability at a high speed, parallel process, etc, to make processing speed and accuracy improve greatly, so it can be applied to real-time control. Performances of the PID-GA hardware controller in different wind disturbance intensities are presented in Table 4. An aircraft is safe to land in wind shear speed at 45 ft/sec, as shown from Figures 8 to 10. Optimal control gains are used in all cases.

Table 4. The results of successful touchdown from using hardware PID controller with EC in wind shear condition

Wind speed	Horizontal velocity (ft/sec)	Landing point (ft)	Aircraft vertical speed (ft/sec)
20	234.6779	721.3338	-2.1559
40	234.6779	730.9726	-2.8532
45	234.6779	825.7964	-2.4126
46	234.6779	850.2211	-2.7592
47	234.6779	720.5599	-2.6772
48	234.6779	874.6542	-2.3217
49	234.6779	890.1424	-2.5478
50	234.6779	912.6667	-2.3328

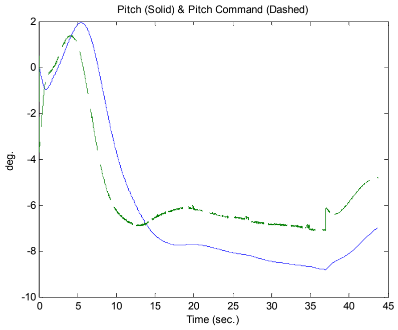

Fig. 8. Aircraft pitch and pitch command

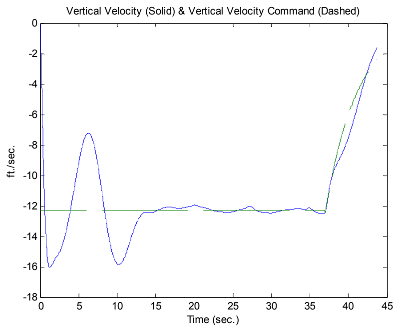

Fig. 9. Aircraft vertical velocity and command

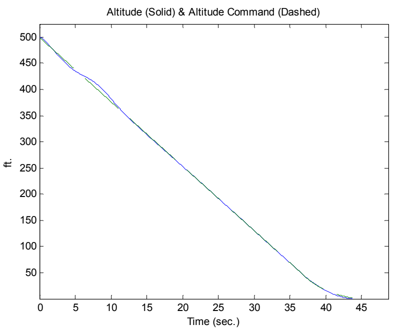

Fig. 10. Aircraft altitude and command

5. CONCLUSIONS

This paper presents an application of a DSP chip for designing a PID-GA controller for aircraft landing system. Compared to the simulation results using Matlab software, the ALS controller is replaced by the DSP controller. On the other hand, evolutionary computation method improves the performance indeed. In EC, we used five kinds of crossover principles to simulate and to compare with their differences. Different wind disturbances are also implemented into the flight simulations. In addition, this study also utilizes the *VisSim software* and *TI C2000 Rapid Prototyper* to develop an embedded control system that uses a DSP controller. Thus, realization of on-line real-time control can be achieved. Simulation results show that the proposed automatic landing controller can successfully expand the aircraft safety envelope to wind shear speed at 50 ft/sec in hardware in the loop simulation and 68 ft/sec in VisSim simulation.

REFERENCES

Adewuya, A.A. (1996). New Methods in Genetic Search with Real-valued Chromosomes, *M.S. thesis*, Dept. of Mechanical Engineering, Massachusetts Institute of Technology.

Berkmann, P. and H.J. Pesch (1995). Abort Landing in Windshear: Optimal Control Problem with Third-Order State Constraint and Varied Switching Structure. *Journal of Optimization Theory and Applications*, **85(1)**, 21-57.

Boeing Publication (2000). Statistical Summary of commercial Jet Airplane Accidents. Worldwide Operations 1959-1999.

Chaturvedi, D.K., R. Chauhan, P.K. Kalra (2002). Application of generalized neural network for aircraft landing control system. *Soft Computing*, **6**, 441-118.

Eshelman, L.J. and J.D. Schaffer (1993). Real-Coded Genetic Algorithms and Interval-Schemata. *Foundations of Genetic Algorithms 2*, 187-202.

Holland, J.H. (1962). Outline for a Logical Theory of Adaptive Systems. *Journal of the Association for Computing Machinery*, **3**, 297-314.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI：University of Michigan Press.

Iiguni, Y., H. Akiyoshi and N. Adachi (1998). An Intelligent Landing System Based on Human Skill Model. *IEEE Transactions on Aerospace and Electronic Systems*, **34**, 877-882.

Ionita , S. and E. Sofron (2002). The Fuzzy Model for Aircraft Landing Control. *Proc. AFSS International Conference on Fuzzy Systems*, 47-54.

Izadi, H., M. Pakmehr and N. Sadati (2003). Optimal Neuro-Controller in Longitudinal Autolanding of a Commercial Jet Transport. *Proc. IEEE International Conference on Control Applications*, **CD-000202**, 1-6.

Jorgensen, C.C. and C. Schley (1991). A Neural Network Baseline Problem for Control of Aircraft Flare and Touchdown. *Neural Networks for Control*, 403-425.

Juang, J.G., H.H. Chang and K.C. Cheng (2002). Intelligent Landing Control Using Linearized Inverse Aircraft Model. *Proceedings of American Control Conference*, **4**, 3269-3274.

Juang, J.G., H.H. Chang and W.B. Chang (2003). Intelligent Automatic Landing System Using Time Delay Neural Network Controller. *Applied Artificial Intelligence*, **17**, 563-581.

Michalewicz, Z. (1992). *Genetic Algorithm + Data Structure = Evolution Programs*, New York：Springer-Verlag.

Nho , K. and R.K. Agarwal (2000). Automatic Landing System Design Using Fuzzy Logic. *Journal of Guidance, Control, and Dynamics*, **23**, 298-304.

Visual Solutions, Inc. (2002). *VisSim User's Guide-Version 5.0.*