Dear Guilhem, Dear Meiko,

Now that I am advancing in my project, I would like to share with you some current results. Let me first give you a brief outline of the structure I would like to pursuit in my final thesis.

- ❒ Introduction:
  -context of the project
  -why should what be done?
- ❒ Identification of important parameters:
  -detection of the parameters with most crucial influence on the touchdown velocity
- ❒ Deterministic velocity calculation(calculating the touchdown velocity neglecting turbulences):
  -Comparison of explicit and implicit model-building [Implementation of the analytical formula and it's evaluation]
  -Introduction of fuzzy inference systems (FIS)
  -Explanation of fuzzy ANFIS
  -Implementation of the ANFIS
- ❒ Integrating turbulences into the consideration:
  -Estimation of turbulences
  -Turbulence contribution to the touchdown velocity
- ❒ Prediction/Advice for go-around maneuver:
  -Unification of the turbulence-estimator and the deterministically calculated turbulence level in order to calculate, if a go - around is necessary
- ❒ Implementation and Evaluation

Contents

Chapter 1

Introduction

Since the early days of passenger aircrafts, there have been countless approaches and investigations to facilitate the pilots task, in order to augment the safety-level in passenger-aviation. Civil aviation has thus been accepted in modern civilization and changed it through offering the possibility of traveling in a fast, reliable and safe way. Regarding the fact, that the air traffic volume doubled in the past 20 years and is supposed to raise significantly in the future 20 years, maintaining the acceptance of civil aviation within the population demands reducing the absolute number of accidents during flight -operations. Examining statistics of aircraft (A/C) accidents, one should recognize, that even today a huge percentage occurs during the final flight phases i.e. approach and landing.

To facilitate further considerations, one can separate accidents in aviation in those which are attributed to flight operations and human factors, and those which are caused by divers technical malfunctions.

Recognizing that the air traffic volume augments, while the number of airports can be assumed to stagnate, the air traffic density and therefore the risk of incidents due to flight operational reasons rise dramatically on frequented airports. Highly frequented airports are usually equipped with costly and sophisticated Instrument Landing Systems (ILS), enabling automatic landings, which from a technical point of view can be considered as safe.

Within the multiple approaches to cope with this bottle-neck effect, the airbus research- and technology department has proposed an alternative landing-technique, allowing passenger aircrafts to land on less frequented aerodromes, without ILS-devices at their disposal, in order to redistribute the air traffic while augmenting safety. During this technique, called Semi - Automatic - Landing, the aircraft is conducted manually until the Flare Initial Point (FIP), where the autopilot is engaged and flares the aircraft to touchdown. It shall be underlined, that the flare is conducted under use of the standard automatic flare law. This concept is based on the statistical fact, that in the long term, autopilots i.e. machines cause less failed landings than human pilots. As a result it is possible to reduce the number of failed landings due to pilot behavior without the need of an ILS device which would be necessary to conduct a fully automatic landing.

In that context failed landings have to be subdivided in different cases. The highly improbable failed landing is the hazard or fatal landing. Whereas the more frequent one is the hard landing, occurring when the A/C's vertical velocity at touchdown is less then 10ft/sec.

Because during fully automatic landings AC's are accurately driven on glide beams by

autopilots, before the flare is initiated, the diversions of the A/C's attitude at FIP are within tiny margins. Consequently the autopilot has to cope with common conditions. On the contrary, in a semi automatic landing, this section is as mentioned before conducted manually. The AC's attitude at FIP will hence vary within broader margins which will more challenge the autopilot during the flare. In certain cases bad A/C conditions at FIP accompanied by demanding environmental parameters such as turbulences or positive runway slopes can evoke hard landings.
In order to further reduce the risk of a hard landing, it is of interest to know the touchdown velocity of an A/C before it touchdowns. With this information it would be possible to advise the pilot to initiate a go-around maneuver, in case a hard landing is predicted. The aim of this work is to propose an assistant for the Airbus A380 aircraft, that foresees the risk of a hard landing. The proposed algorithm shall give an advice to the pilot in the moment the flare is initiated.

Chapter 2

Crucial Parameters

The advice to initiate a go - around, has to be based on a predicted touchdown - velocity ($v_{z\ imp,pr}$). For this prediction, it is necessary to detect the parameters which the A/C's touchdown - velocity. Therefor, an analysis of the final flight phase before touchdown will be conducted within this chapter. The aim is thus, to find the parameters $x_1 \rightarrow x_n$ in order to afterwards formulate a relation:

$$v_{z\ imp} = v_{z\ imp}(x_1, x_2, \cdots, x_n) \tag{2.1}$$

2.1 Analysis of aircraft's flare and touchdown

From a general point of view, the motion of an aircraft is a 3 - dimensional problem. Nevertheless, the final flight phase (flare) can for simplification be considered as a symmetrical and thus 2 - dimensional flight maneuver. That is why the following considerations will only refer to effects in the xz - plane.

The flare maneuver targets to smoothly reduce the vertical velocity before touchdown. At the beginning of the flare, i.e. at the flare initial point (fip), the aircraft descends at a initial rate($v_{z\ fip}$). In the moment the aircraft touchdowns, the vertical velocity is reduced to a lower rate ($v_{z\ imp}$) which is usually approximately -2.5ft/sec. To achieve this reduction in the vertical velocity, the aircraft follows a curved vertical flight pass called flare trajectory.

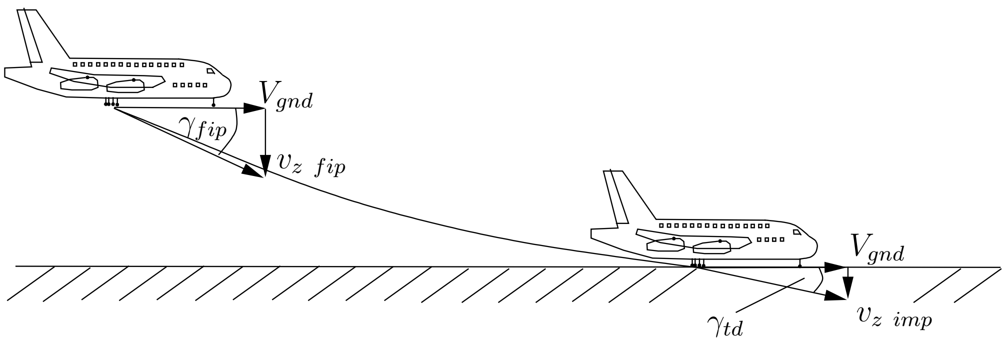

Figure 2.1: flare maneuver

During the semi automatic landing, the aircraft is automatically conducted on this trajectory. Therefor a pre - determined reference trajectory is calculated from the autopilot at the beginning of the flare. Since Airbus aircrafts are controlled through load - factor commands (n_z), the reference trajectory, a polinomial of 4^{th} order, exhibits the shape:

$$n_z(t) = a^4t + b^3t + c^2t + dt + e. \tag{2.2}$$

The parameters of the n_z-flare trajecotry $a \rightarrow e$ depend on the ground speed (V_{gnd}). For simplification, it is assumed, that the ground speed remains constant during the flare.

$$a, b, c, d, e = a, b, c, d, e(V_{gnd}) \tag{2.3}$$

The n_z-flare trajectory can be integrated, in order to receive a profile that describes the vertical velocity v_z during the flare :

$$v_z(t) = a^5t + b^4t + c^3t + d^2t + et + f. \tag{2.4}$$

The constant of integration (f) is the initial descent rate at the FIP ($v_{z\ fip} = v_z(t_{flare} = 0) = v_{z0}$). Consequently one obtains:

$$f = v_{z0} = V_{gnd} \cdot tan(\gamma_{fip}) \rightarrow f = f(V_{gnd}, \gamma_{fip}) \tag{2.5}$$

By means of the equations 2.3 $\rightarrow$ 2.5 one can see, that the impact velocity ($v_{z\ imp}$) is affected from the initial flight path angle (γ_{fip}) and the ground velocity (V_{gnd}):

$$v_{z\ imp} = v_{z\ imp}(\gamma_{fip}, V_{gnd}) \tag{2.6}$$

Equation 2.4 assumes a even runway. In reality, there are frequently airports, where the runway is not even - but inclined. This inclination of the runway, called runway - slope (γ_{rwy}) is usually indicated in percent (%). Depending on it's prefix, the inclination can raise or reduce the touchdown velocity.

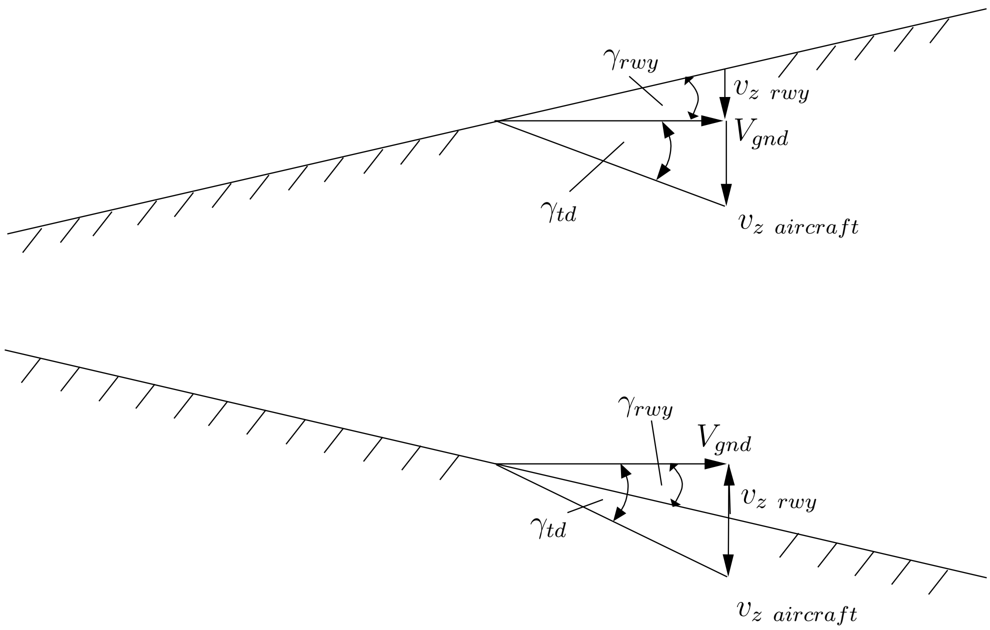

Figure 2.2: runway slope

If the autopilot of an aircraft would not consider the runway slope, the touchdown - velocity would be the sum of the vertical velocity of the airctraft at touchdown ($v_{z\ aircraft}$) and the component called ($v_{z\ rwy}$).

$$v_{z\ imp} = v_{z\ aircraft} + v_{z\ rwy} \tag{2.7}$$

The flare controller of the A380 is able to compensate a huge part of this effect, because the radar altimeter, which is activated during the flare, detects a runway slope through differentiation of the measured altitude. Hence it is possible to adapt the flare to local environmental conditions. Nevertheless a small effect can remain on the vertical velocity. That is why equation 2.6 is extended:

$$v_{z\ imp} = v_{z\ imp}(\gamma_{fip}, V_{gnd}, \gamma_{rwy}) \tag{2.8}$$

To visualize the impact of the mentioned parameters ($\gamma_{fip}, V_{gnd}, \gamma rwy$), the touchdown velocity (obtained from simulations using the Airbus - software "SIMPA") is plotted against the variation of each parameter, while freezing the other parameters to default values. The aircraft mass is further set to a default value of 390t. To further accent each parameter's influence, the dynamic atmosphere (turbulences) have been disengaged in the simulation.

Notice:
The ground - speed (V_{gnd}) is the sum of the airspeed and the longitudinal wind velocity:

$$V_{gnd} = V_A + V_{w\ long} \tag{2.9}$$

As it is due to the structure of the software (SIMPA) easier to vary the wind velocity instead of the ground speed, the variation of the ground speed is replaced through a variation of the wind velocity ($V_{w\ long}$). A negative prefix of the wind velocity corresponds to *tailwind, whereas a positive prefix corresponds to headwind.*

The default values are:

- ❒ $\gamma_{fip_d} = -3°$
- ❒ $\gamma_{rwy_d} = 0\%$
- ❒ $v_{w\ long_d} = 0kt$

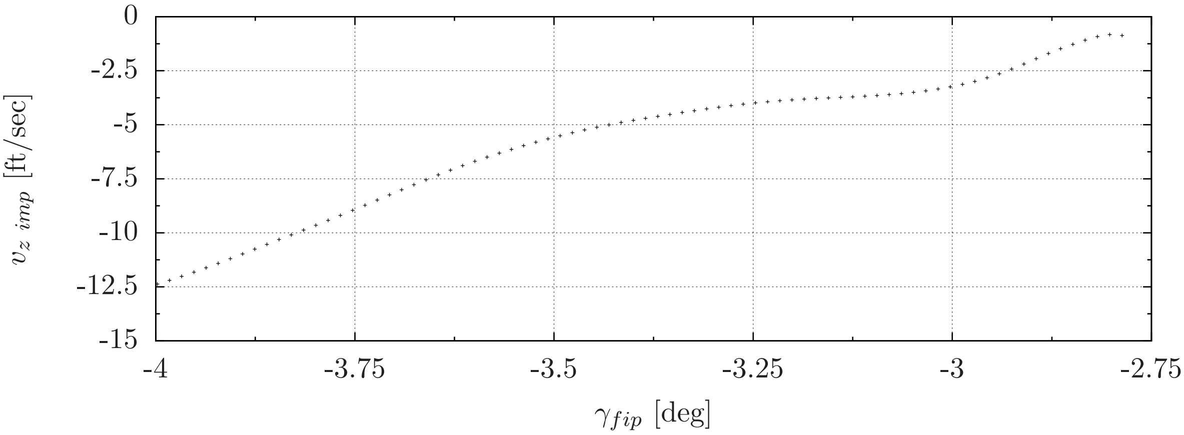

Figure 2.3: influence of the initial flight pass angle

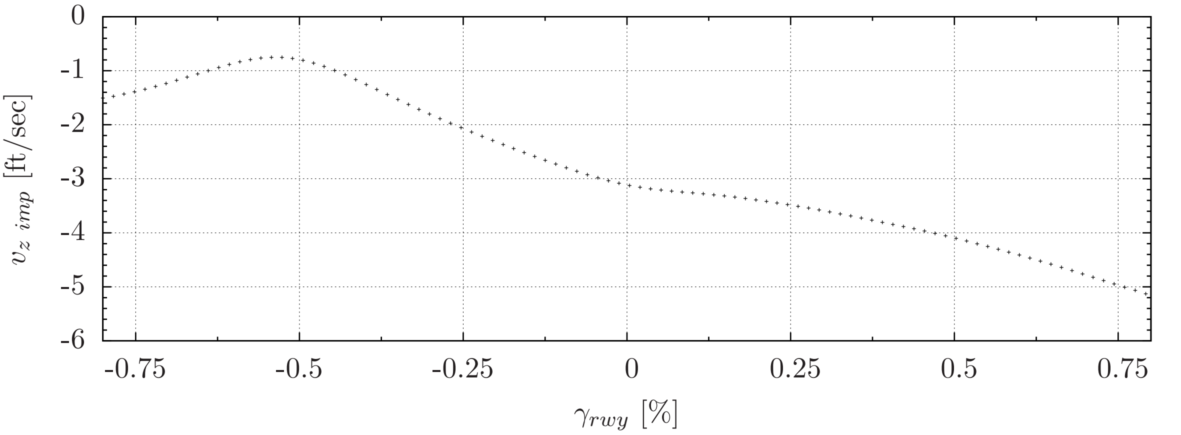

Figure 2.4: influence of the runway slope

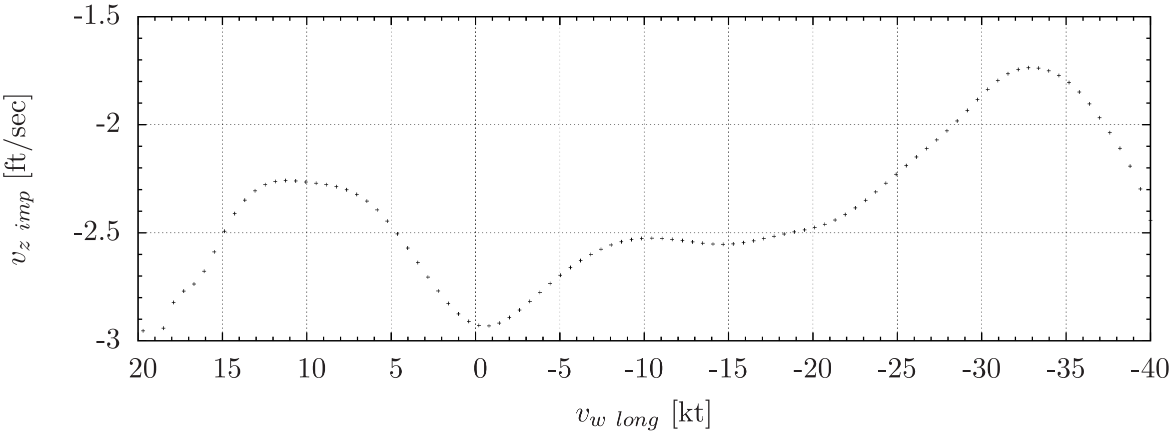

Figure 2.5: influence of the longitudinal wind velocity

Regarding fig.2.1 → fig.2.3 it is obvious, that within the observed variations, the most sensitive parameter is the initial (fpa). The steeper the fpa is, the harder the aircraft touchdowns. The parameter "f" of equation 2.4 is thus huge, compared to the rest of the term.
Fig.2.2 underlines that in spite of the runway slope compensation, an effect in the touchdown velocity remains. A positive runway slope augments the touchdown velocity - whereas negative inclinations result in smaller touchdown velocities.
Fig.2.3 depicts the characteristic impact of the wind, on the A380's touchdown velocity.

2.2 Influence of turbulences

Besides the parameters, found in the previous section, it is important to comprise, that the dynamic atmosphere has an impact on the touchdown velocity of an aircraft. It is obvious, that turbulences, i.e. accelerations, especially those in the vertical axis, impact the touchdown velocity.
Notice that it is within this study for simplification assumed, that the influence of horizontal turbulences is negligible, compared to turbulences in the vertical axis.
On the one hand, turbulences have an impact, in the moment the aircraft touchdowns (t_{td}). Depending on the orientation of the vertical wind velocity at (t_{td}), their influence can, comparable to the runway slope, increase or disarm the situation. On the other hand, turbulences continuously influence the flare, through disturbance of the flare trajectory. They thus have as well an influence on the initial fpa (γ_{fip}).

Contrary to the parameters, elucidated in the last section, it is quiet intricate to deal with turbulences due to their random character. The runway slope for instance is a static parameter with a repeatable and therefor easily predictable influence. FIGURE XXX illustrates, that one turbulence level can evoke different impact velocities. It is therefor worth, having a view on possibilities to cope with turbulences.

There are attempts, trying to describe turbulent fluid flows in a numerical way. Nonetheless, in praxis it is due to countless nonlinear and almost inapprehensible effects, unpromising and therefor uncommon to describe turbulences this way. That is why they are usually described and expressed using statistic methods.

Considering turbulences as a gaussian - distributed and random process, like it is defined by aviation authorities, the most meaningful parameter is the turbulence intensity. The intensity σ indicates the average deviation from the mean wind velocity. It is a factor, which depends on several environmental parameters, such as the roughness of the ground, the thermal gradient, the altitude and the mean wind velocity. The definition of the intensity is given through following formula:

$$\sigma = \sqrt{\frac{1}{t_o} \int_{t_0}^{t_o} v_w^2(t) dt} \qquad (2.10)$$

where t_o is the time of observation and v_w is the vertical wind velocity.

2.3 Parameter Classification

In the previous reflections, the influence of the 4 parameters has been elucidated. It has on top of this been illustrated, that turbulences evoke a random influence on the touchdown - velocity, whereas the remaining 3 parameters have a deterministic influence. The indisputable discrepancy in the parameters comportment has to be included in the prediction - algorithm. Because the impact of turbulences is randomly, the output of the touchdown - velocity predictor cannot be a crisp value. To underline this obvious fact Fig(2.6) shall be considered. By means of the deterministic parameters ($\gamma_{fip}, \gamma_{rwy}, v_{w\ long}$) and a known relation between them and the touchdown - velocity, it would be possible to detect, if the aircraft would touchdown with less or more than -10ft/sec (in a static atmosphere). In case the estimated velocity would be less than -10 ft/sec a go - around - advise can be triggered.

In a true environment, turbulences have to be involved. As emphesized multiple times, turbulences reduce or raise the touchdown - velocity. Nevertheless they augment the probability of a hard(er) landing. Depending on the turbulence intensity, the diversions, and thus the probability of a harder or smoother touchdown - velocity will vary. Regarding Fig.2.6 the aircraft would touchdown at vertical velocity of -7ft/sec. Consequently there remaining tolerance ($\Delta v_{z\ imp}$) to a hard landing is -3ft/sec. Depending on the level of the turbulence, the probability p($\Delta v_{z\ turb}$) = $\Delta v_{z\ imp}$ would vary. By means of a statisitcal relation, between the turbulence level, it is possible to calculate the probability of a hard landing.

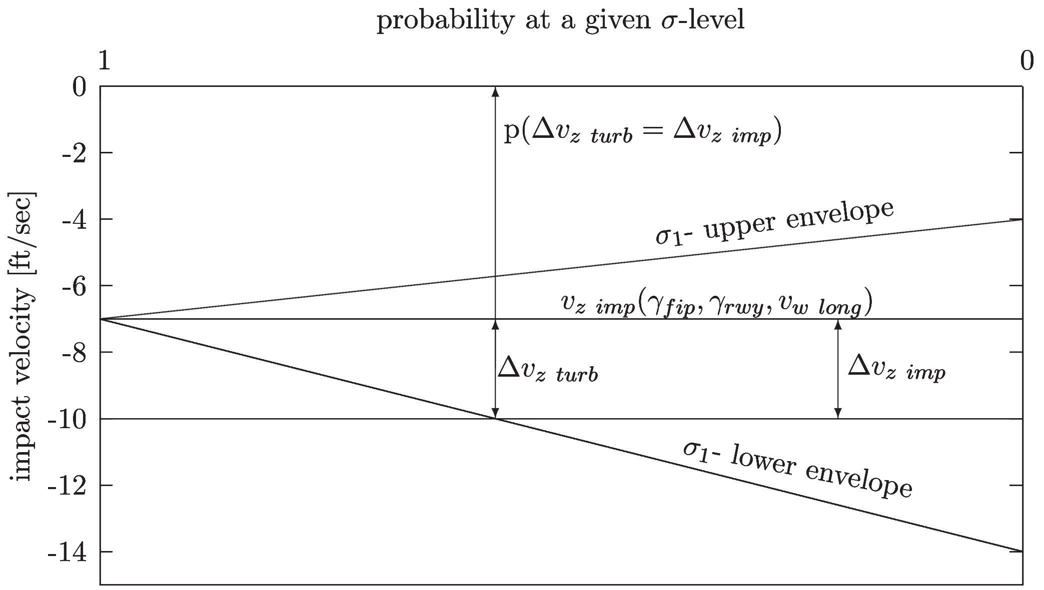

Figure 2.6: parameter characteristics

Fig.2.6 emphesizes, that the design of the go around advisory can only be based on a probability of a hard landing. Moreover the deterministic and the stochastic parameters have first to be computed seperately.(see Fig.2.7)

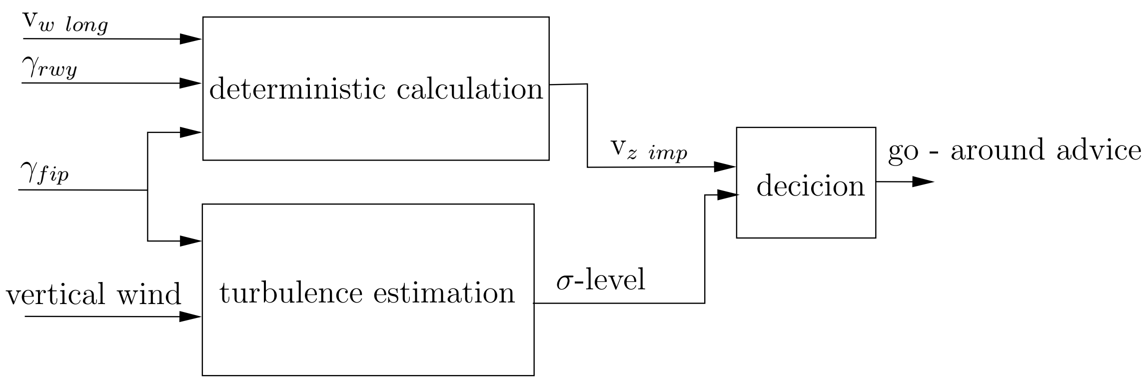

Figure 2.7: go - around - advice

As a conclusion of this chapter, 3 tasks have to be done, in order to design the algorithm. First it is necessary to find a model, which describes the relation between the deterministic parameters and the touchdown velocity in a static environment.

$$v_{zimp\ deterministic} = v_{zimp\ deterministic}(\gamma_{fip}, \gamma_{rwy}, v_{w\ long}) \tag{2.11}$$

Secondly it is evident, to design an algorithm that on - board estimates the turbulence level before, the advice is triggered i.e. during the manually conducted part of the landing. Afterwards a relation between the turbulence level and the probability of a diversion in the touchdown - velocity has to be found. Finally the deterministic part and the turbulence - part have to be coupled.

Chapter 3

Deterministic velocity calculation

3.1 Comparison of explicit and implicit model-building

In general, one can distinguish between two ways of deriving mathematical models. The first opportunity is to build a model, based on knowledge.

Within numerous variants to express knowledge in a technical environment, physical statements are the most common. Such models are described by phenomena which are expressed in differential equations. Due to the reason that the functionality of a system has to be comprehended, before it can be formulated in physical terms, the model gets automatically transparent i.e. comprehensive. Putting it into other terms, one can consider the model as a "readable" box.

Another way to build a knowledge - based model, is the Fuzzy Inference System (FIS). FIS are systems, using the theory of fuzzy sets and fuzzy logic. The driving idea of a FIS is the use of expert - knowledge (knowledge emulating the behavior of a human being), expressed in simple rules. Due to the rules shape and the model assembly, the system gets easily comprehensible and structured. Hence such a system can also be considered as "readable". On top of this a FIS contains linear dependencies of each rule on the input variables. The implementation and integration of a FIS into an aircraft is therefor quiet simple in comparison to other methods.

The transparency of pure, knowledge - based models is accompanied disadvantages. Building a knowledge - based system, affords the total comprehension of the system. Dealing with ordinary and even more demanding systems, it is possible to get powerful models, since the afforded knowledge is easily available. In extremely sophisticated systems, like the A/C response during flare, this kind of modeling would be time - consuming and therefore less competitive. The reason is the interaction between several nonlinear effects such as the ground effect (see the last report I sent you) before touchdown.

A different opportunity to derive a model, is the data - based approach. In contrast to models built upon explicit knowledge, experimental data gets the driving source. Thus one can say the knowledge is derived implicitly. The main advantage is obvious: Following this approach affords less explicit knowledge. On condition of possessing the right experimental data, it is possible to design models of systems which can only hardly be modeled pursuing an explicit - knowledge based approach.

In this context, neural networks (NN), whose basic principles are leaned to the human brain, have gained in importance during the past decades by offering the possibility to cope with highly sophisticated systems. On top of that, they have the unique ability of learning, and hence extracting knowledge. Like the human brain a neural network consists of interconnected entities called neurons. Through the parallel use of these cells, a network gains the capability of performing extremely fast. In order to transmit signals individually from one neuron to another, the cells are connected by gains. Each of the neurons has several inputs and one output. The output can either embody the input to other neurons - or the final output of the network. Neural networks learn by iterative adjustment of these gains. To fulfill this task they depend on data, usually consisting of related input and output pairs.

Besides this advantage (learning), implementing a neural network means making compromises concerning the model transparency. The cause of this problem is lying in the fact, that it is nearly impossible to track which neuron had which contribution to the sysmtem's output. As a consequence, the character of a neural network is comparable to a black box. To put it into different terms: the model is built upon implicit knowledge that cannot be extracted once the model is build. Only the knowledge of a NN involving a small number of layers, can be extracted reasonably. Since in general NN contain many layers and a huge number of parallel nodes, the attempt to understand a system by putting the evidence of all nodes in one formula fails. There are certainly different target environments where system transparency can be totally neglected. Contrary to these surroundings safety critical target environments like applications in aviation always afford traceable, comprehensible models. That is one reason why the application of pure neural networks in aviation - and thus in the deterministic prediction of the touchdown velocity - cannot be taken into consideration.

In the previous reflections it has been illustrated, that both regarded opportunities to build a model, the knowledge - based and the data - based, have advantages as well as expulsing deficits. The transparency of models, based on explicit knowledge usually causes a higher development effort. Models derived from implicit knowledge can be build automatically. Nevertheless it is intricate to understand their contents, once they are build.
It becomes by consequence hard to calculate the touchdown velocity using only one of these approaches. The potential rising by unifying benefits (learning, from data & reducing the design effort while maintaining a grade of transparency)of both options, directs to the emergence of so called hybrid intelligent systems (HIS), algorithms using in parallel different methods. In general, subfields, applicable to HIS are "probabilistic reasoning", "fuzzy logic" and "neural networks". That is why an analytical, knowledge - based model could not be build by means of HIS.

Among the huge amount of HIS - algorithms, there is one especially dedicated to calculate the touchdown velocity. This technology called "Adaptive Neuro Fuzzy Inference System" (ANFIS) enables building a transparent model - based on data. To put it into literature terms citing Negnewitsky :"Combining German mechanics with Italian love". The main idea standing behind ANFIS, is to automatically derive a traceable model from implicit knowledge - hidden in a set of related input - output data- by means of a neural network. The extraction is done buy building a Fuzzy Inference System under use of an adaptive neural network (ANN). To better comprehend the functionality of ANFIS, it is worth having a look on how each of the methods it includes work individually.

3.2 Fuzzy Inference Systems (FIS)

The principles of FIS are illustrated - accompanied by a practical example i.e. a car distance control (CDC) as depicted in Fig.3.1.

Figure 3.1: Car Distance Control

Example introduction:
The CDC is a system, designed to to maintain a safe distance to forward driving traffic - in case the driver exceeds dangerous thresholds. It thus targets the avoidance of so called rear - end collisions. The CDC's inputs are the distance x_1, measured in m by a on - board radar - device which is fixed in the front of the car and the relative velocity to the forward traffic x_2 given in km/h. The assistant's output y is the commanded brake - effort given in %.

Fuzzy sets, the basics of each FIS, have been invented in 1965 by Lafti Zadeh with the intention of extending classical crisp sets. In contrast to a crisp set i.e. a set of Boolean values, the fuzzy set is multi - valued. In the classical Boolean theory an element x can either belong to a set U - or not. That is why the set is considered being crisp. By consequence of the sharp boundary, each member of the set will have a value of 1 and each element outside the boundary gets the value 0.

$$f_U(x) : X \to 0, 1 \tag{3.1}$$

$$f_U(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A. \end{cases} \tag{3.2}$$

In contrast, within a fuzzy set, the boundary between being a member and not being part of the set, is widened. This is done through allocation of a degree of membership to each element within a given range.

$$\mu_U(x) : U \to [0, 1] \tag{3.3}$$

where:

$$\mu_U(x) = 1 \text{ if x is totally in U;} \tag{3.4}$$

$$\mu_U(x) = 0 \text{ if x is not in U;} \tag{3.5}$$

$$0 < \mu_U(x) < 1 \text{ if x is partly in U.} \tag{3.6}$$

The term logic represents the art of expressing knowledge by transforming crisp inputs in membership degrees and treating them through application of simple rules. Basically there are 2 types of FIS (Mamdani and Sugeno) differing in their rules. Due to the fact, that Sugeno systems are computationally more efficient and adapted to ANFIS, Mamdami FIS will not be considered furhter.

In general, the flow of computation inside a FIS is done in different steps:

- fuzzification
- rule evaluation
- deduction of output

Fuzzification:

Within the fuzzification, the crisp input - values $x_1 \to x_i$ of a given domain are transformed into grades of membership, ranging from 0 to 1. For a domain X a fuzzy set A over X is defined by a membership function $\mu_A(x) \to [0, 1]$ as mentioned above. The root idea of this step, is the introduction of linguistic variables. Each linguistic variable is introduced through an expert and relates one rated attribute to each of the fuzzy sets. An expert is somebody who is able to rate the importance of an input due to his knowledge. This way, each input - variable can address at least one attribute, depending on it's crisp input value. As will be seen later, the degree of membership to one attribute, influences each rule that contains this attribute, and thus the system's output.

Example part1:
Within the CDC, each input shall correlate with 2 linguistic variables i.e with 2 membership functions (FIG). The distance x_1 shall correlate with the attributes "NEAR" &"FAR". Whereas the "velocity x_2 shall correlate with the attributes "SLOW" & "FAST". Since the shape of the membership function does not influence the FIS principle, it is for simplification set trapezoidally. At the moment, the car is observed, it holds a distance of 400 m to the forward traffic while driving at a relative velocity of 90km/h. The crisp input "distance" is rated by an expert to the mentioned functions as 0.375 NEAR and 0.125 FAR whereas the "speed" is rated to 0.2 SLOW and 0.8 FAST.

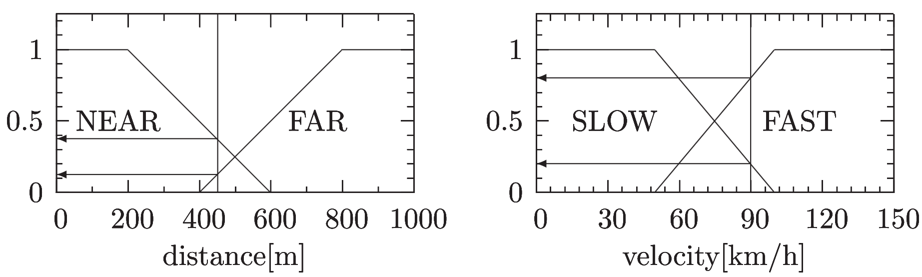

Figure 3.2: Fuzzification

Rule evaluation and deduction of system's output:

In the next computational step of a Sugeno FIS, the crisp inputs of the system are applied on each rule they belong to. In Sugeno FIS the rules consist of two parts. The premise, representing a condition, which demands the membership degree of one input:

$$\text{If } f(x_1 is A_1, ..., x_k is A_k) \text{then } y = g(x_1, ...x_k) \tag{3.7}$$

And the rules consequence, which is carried under the condition of a true premise. In Sugeno FIS it is a linear function depending on the crisp input values.

$$y_i = p_0 + p_1 x_1 + ... + p_k x_k \tag{3.8}$$

For every rule the consequence y^i is represented by the function g^i in the consequence:

$$y^i = g^i(x_1^i, ..., x_k^0) = p_0^i + p_1^i x_1^0 + ... + p_k^i x_1^0 \tag{3.9}$$

By consequence, a set of i values is obtained. In order to unify all values to get one representative system output, the next step called inference has to be conducted. The system's output y is inferred from the i implications as the average from all weighted y^i:

$$y = \frac{\sum |f^i * y^i|}{\sum |y^i|}. \tag{3.10}$$

The firing strength, that weights the contribution of each independent rule to the final output, is calculated by means of the fuzzy "AND" - operator in case of multiple antecedents in one rule:

$$f^i = \mu(A_1) \wedge \mu(A_2) \wedge ... \wedge \mu(A_n) \tag{3.11}$$

One can also consider the firing strength as a truth value. Only if all membership degrees in the premise of one rule are huge, the contribution of this rule gains in importance to the final output of the system.

Example part2:
In order to raise performance in every thinkable situation, the CDC shall obtain the following 4 rules.

$$\begin{aligned}
&RULE1 : \text{If distance "FAR" and velocity "HIGH", then:} && y^{(1)} = x_1 + x_2 \\
&RULE2 : \text{If distance "FAR" and velocity "SLOW", then:} && y^{(2)} = 0.5x_1 + 0.5x_2 \\
&RULE3 : \text{If distance "CLOSE" and velocity "HIGH",then:} && y^{(3)} = 3x_1 + 3x_2 \\
&RULE4 : \text{If distance "CLOSE" and velocity "SLOW",then:} && y^{(4)} = 0.25x_1 + 0.25x_1
\end{aligned}$$

By means of knowledge, how a driver would react in each situation, it is possible to generate the linear equations of the consequence. Rule No 1's consequence equation includes the expert knowledge, that while driving FAST and FAR away from forward traffic, a driver would decelerate the car gently. Whereas a driver in a car -FAR away, at low velocity- would almost not decelerate the car. That is why small coefficients are chosen for this consequence equation. The third rule's consequent expresses the necessity to severely reduce the velocity while driving FAST, directly behind the forward car. The coefficients are therefore enlarged in comparison to the previous ones. The last consequent equations contains the smallest coefficients since it is not necessary to slower a car which is driving at low velocity behind forward traffic.

Fig.1.3 depicts the rule evaluation and it's inference. The first rule's firing strength is reduced by the input "distance", since the chosen membership function judges a distance of 400m as "a bit" FAR. Hence the contribution of the first rule to the final out is small. The firing strength of the other rules is extracted the same way. The individual outputs of all rules are calculated as mentioned above, through applying the crisp input values on the linear predefined equations.

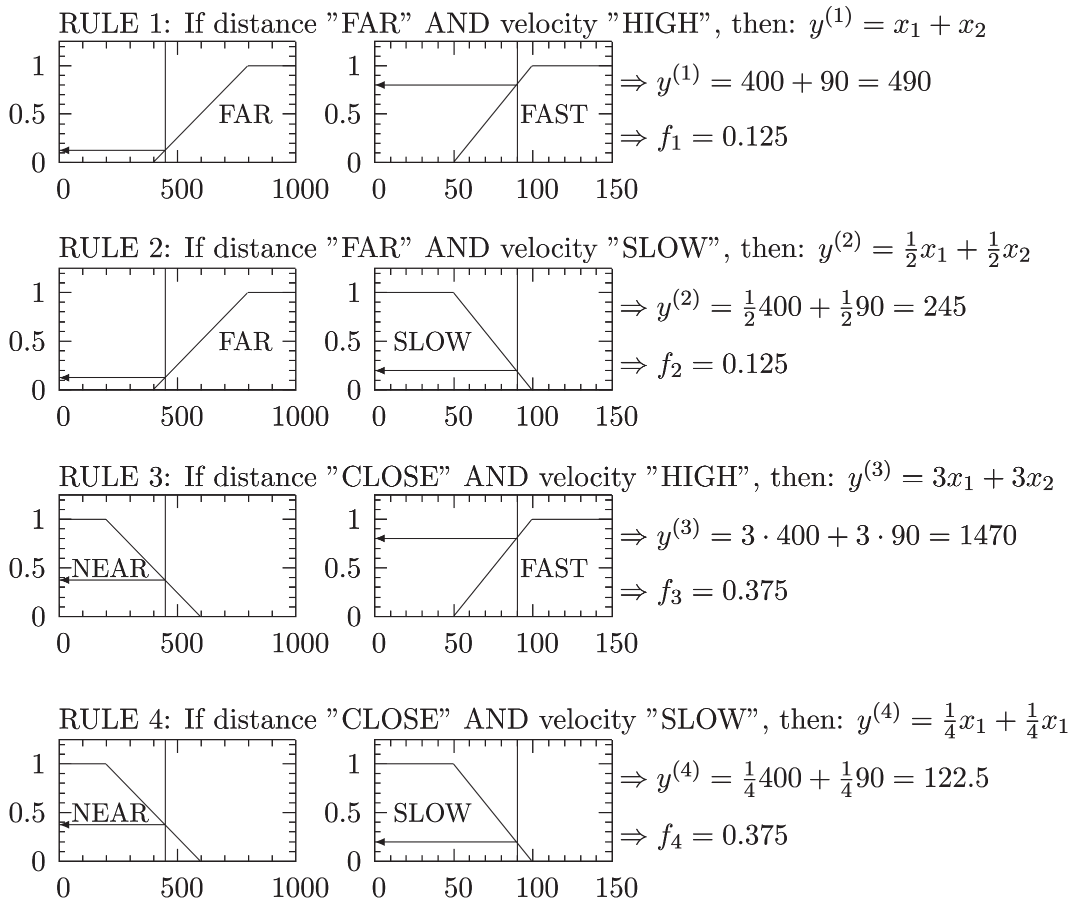

Figure 3.3: rule-evaluation and inference

After the calculation of the 4 individual rule outputs and their firing strengths, the total brake effort is computed as mentioned above:

$$y = \frac{0.125 \cdot 490 + 0.125 \cdot 245 + 0.375 \cdot 1470 + 0.375 \cdot 122.5}{490 + 245 + 1470 + 122.5} = 0.296 = 29,60\% \qquad (3.12)$$

3.3 Adaptive Neuro Fuzzy Inference System (ANFIS)

In the last sections, the benefits and principles of Fuzzy Inference Systems have been illustrated. Nevertheless it would be time - consuming and extremely intricate, to build a fuzzy model - that calculates the vertical touchdown velocity - by means of a human expert.
It has on top of this been emphasized in previous considerations, that through the hybrid use of FIS and neural networks unified in the ANFIS algorithm, it becomes possible to automatically build FIS. In other terms it becomes possible to automatically build implicit and transparent models. Due to the reason, that it is worth to comprehend ANFIS, before implementing it, it's basic functionality shall be regarded in the following.

3.3.1 Assembly of ANFIS

Generally ANFIS extracts fuzzy rules of the type:

$$\text{If } f(x_1 \; is \; A_1, ..., x_k \; is \; A_k) \text{ then } y = g(x_1, ...x_k)$$

from a given set of related input-output data.
Fig.3.4 illustrates the basic assembling of ANFIS, which has been introduced by Jang in terms of a six - layer feed - forward neural network. In order to reduce the complexity, the number of inputs in the figure is set to 2. Nevertheless it has to be outlined, that ANFIS could theoretically handle a system with countless inputs.

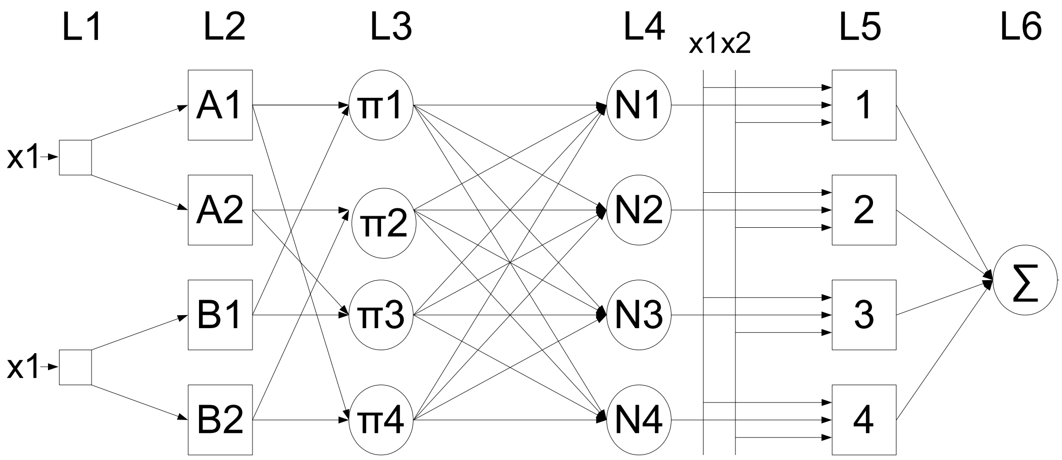

Figure 3.4: Structure of ANFIS

As in the CDC example, each input correlates with 2 fuzzy sets (A_1, A_2, B_1, B_2). By consequence the 4 potential rules are given with:

$$
\begin{aligned}
&RULE1: \text{If } x_1 \text{ is } A_1 \wedge x_2 \text{ is } B_1, \text{ then:} && y^{(1)} = k_{10} + k_{11}x_1 + k_{22}x_2 \\
&RULE2: \text{If } x_1 \text{ is } A_1 \wedge x_2 \text{ is } B_2, \text{ then:} && y^{(2)} = k_{20} + k_{21}x_1 + k_{22}x_2 \\
&RULE3: \text{If } x_1 \text{ is } A_2 \wedge x_2 \text{ is } B_1, \text{ then:} && y^{(3)} = k_{30} + k_{31}x_1 + k_{32}x_2 \\
&RULE4: \text{If } x_1 \text{ is } A_2 \wedge x_2 \text{ is } B_2, \text{ then:} && y^{(4)} = k_{40} + k_{41}x_1 + k_{42}x_2
\end{aligned}
$$

The rule consequences are given in a polynomial form with k_{10} to k_{42} as tuning parameters. These tuning parameters are necessary to allocate a degree of freedom to the rules consequence - equations, when training the system.
Since it is important to address the cells of the ANFIS, the chosen index - system is shortly described: As mentioned before, the "exponent" of the $y^{(i)}$ denotes the layer. The indexes of the inputs $x_{(i)}$ and the first indexes of the tuning parameters denote the number of the input. $O_m^{(k)}$ for instance would be the m-th neuron of the k-th layer. By consequence m is the number of input variables whereas n denotes the number of neurons inside the rule - layer.
As can be seen in Fig.3 the NN contains round and rectangular neurons. Neurons with a rectangular shape are adaptive- which means their parameters can be tuned, whereas Neurons with a round shape are static.
The first layer of the ANFIS-network, the Input layer, passes the crisp inputs x_1 and x_2 in order to direct them to the membership neurons in Layer 2. To express the desired membership functions, the neurons of this layer have to be adaptive i.e they have to contain free parameters. A commonly used membership function shape is the bell:

$$O_i^{(2)} = y_i^{(2)} = \frac{1}{1 + (\frac{x_i^2 - a_i}{c_i})^{2bi}} \tag{3.13}$$

The parameters a_i, b_i and c_i are the degree of freedom given to the bell-function. By means of varying the first parameter a_i it is possible to influence the center of the bell, whereas the parameter b_i and c_i impact the slope and the width of the bell-shaped membership-function.
Each neuron (i) of the third layer calculates the firing strength i.e. the truth value of the rule it is standing for via multiplication of the membership degrees of it's antecedent branch:

$$O_i^{(3)} = y_i^{(3)} = w_i = \mu_{A_i}(x_1) \wedge \mu_{B_i}(x_2) \tag{3.14}$$

In the 4th layer each rule's firing strength is normalized through calculating the ratio between the contribution of each rule's firing strength to the total firing strength:

$$O_i^{(4)} = y_i^{(4)} = \overline{w_i} = \frac{w_i}{\sum\limits_{i=1}^{n} w_i} \tag{3.15}$$

Layer 5 can be seen as the deffuzification layer. Neurons in layer 5 receive on the one hand the output of layer 4 i.e. the normalized firing strength, and at the same time the crisp input values x_1 and x_2 of the system. By means of these entries, each node in layer 5 computes the contribution of the rule it represents, towards the whole output:

$$O_i^{(5)} = y_i^{(5)} = \overline{w_i}(k_{i0} + k_{i1}x_1 + k_{i2}x_2) \tag{3.16}$$

In the single neuron of layer 6 the summation of all weighted partial-outputs is computed and thus the total output is given through:

$$O^{(6)} = y^{(6)} = \sum_{i=1}^{n} \overline{w_i}(k_{i0} + k_{i1}x_1 + k_{i2}x_2). \tag{3.17}$$

3.3.2 How ANFIS learns

The algorithm enabling ANFIS to learn fuzzy - rules from a given data set is subdivided in 2 passes. In one training epoch, consisting of a forward and a backward pass, it uses the least - squares estimation (lse) in the forward branch, and the gradient-decent-method (gde) in the reverse branch.
Before the training is launched, an initialization is conducted by assigning automatically initial activation parameters (a,b,c) to the neurons of the third layer. The parameters are chosen in order to equally divide the span of the bell - functions over the range of one input - vector x_i, and at the same time in order to enable sufficient overlapping to the widths and slopes of the bell - functions. a,b,c are further the tuning - parameters allocated to the fuzzification i.e the second layer of the network.

Forward pass, least - squares estimation

In the forward pass of one epoch, a set of q input - vectors is applied on the NN and the outputs of each layer are calculated layer by layer. Within this forward - cycle, the rules consequent parameters k_{10} to k_{nm} are adjusted, i.e. compared using the least - squares estimation. By means of this computation, a vector Y_d consisting of a set of q linear equations is obtained:

$$\left.\begin{aligned}
y_d(1) = {} & \overline{w_1}(1)[k_{10} + k_{11}x_1(1) + k_{12}x_2(1) + ... + k_{1m}x_m(1)] \\
& + \overline{w_2}(1)[k_{20} + k_{21}x_1(1) + k_{22}x_2(1) + ... + k_{2m}x_m(1)] + ... \\
& + \overline{w_n}(1)[k_{n0} + k_{n1}x_1(1) + k_{n2}x_2(1) + ... + k_{nm}x_m(1)] \\
y_d(2) = {} & \overline{w_1}(2)(k_{10} + k_{11}x_1(2) + k_{12}x_2(1) + ... + k_{1m}x_m(2)] \\
& + \overline{w_2}(2)[k_{20} + k_{21}x_2(2) + k_{22}x_2(2) + ... + k_{2m}x_m(2)] + ... \\
& + \overline{w_n}(2)[k_{n0} + k_{n1}x_2(2) + k_{n2}x_2(2) + ... + k_{nm}x_m(2)] \\
& \vdots \\
y_d(p) = {} & \overline{w_1}(p)[k_{10} + k_{11}x_p(p) + k_{12}x_2(p) + ... + k_{1m}x_m(p)] \\
& + \overline{w_2}(p)[k_{20} + k_{21}x_p(p) + k_{22}x_2(p) + ... + k_{2m}x_m(p)] + ... \\
& + \overline{w_n}(p)[k_{n0} + k_{n1}x_p(p) + k_{n2}x_p(2) + ... + k_{nm}x_m(p)] \\
& \vdots \\
y_d(q) = {} & \overline{w_1}(q)[k_{10} + k_{11}x_1(q) + k_{12}x_2(q) + ... + k_{1m}x_m(q)] \\
& + \overline{w_2}(q)[k_{20} + k_{21}x_2(q) + k_{22}x_2(q) + ... + k_{2m}x_m(q)] + ... \\
& + \overline{w_n}(q)[k_{n0} + k_{n1}x_2(q) + k_{n2}x_p(q) + ... + k_{nm}x_m(q)]
\end{aligned}\right\} \tag{3.18}$$

Putting it in matrix notation, the output is hence:

$$Y_d = \begin{bmatrix} y_d(1) \\ y_d(2) \\ \vdots \\ y_d(p) \\ \vdots \\ y_d(q) \end{bmatrix} = \mathbf{A}k \tag{3.19}$$

with

$$\underbrace{\mathbf{A} = \begin{bmatrix} \overline{w_1}(1) & \overline{w_1}(1)x_1(1) & \dots & \overline{w_1}(1)x_m(1) & \dots & \overline{w_n}(1) & \overline{w_n}(1)x_1(1) & \dots & \overline{w_n}(1)x_m(1) \\ \overline{w_1}(2) & \overline{w_1}(2)x_1(2) & \dots & \overline{w_1}(2)x_m(2) & \dots & \overline{w_n}(2) & \overline{w_n}(2)x_1(1) & \dots & \overline{w_n}(2)x_m(2) \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ \overline{w_1}(p) & \overline{w_1}(1)x_1(p) & \dots & \overline{w_1}(p)x_m(p) & \dots & \overline{w_n}(p) & \overline{w_n}(p)x_1(p) & \dots & \overline{w_n}(p)x_m(p) \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ \overline{w_1}(q) & \overline{w_1}(1)x_1(q) & \dots & \overline{w_1}(q)x_m(q) & \dots & \overline{w_n}(q) & \overline{w_n}(q)x_1(q) & \dots & \overline{w_n}(q)x_m(q) \end{bmatrix}} \tag{3.20}$$

and

$$k = [k_{10}\ k_{11}\ k_{12}\ \dots\ k_{1m}\ k_{20}\ k_{21}\ k_{22}\ \dots\ k_{n0}\ k_{n1}\ k_{n2}\ k_{nm}]^T \tag{3.21}$$

As mentioned before the vector k includes the unknown consequent parameters of the rules:

$$k = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T y_d \tag{3.22}$$

Reverse - pass, backpropagation

Whereas in the forward pass the parameters k_{10} to k_{nm} are tuned, the backward pass attends to gain adequate parameters of the antecedents (a,b,c) through application of the gradient - decent method. This procedure is illustrated by considering the parameter a of the neuron A_1^2 i.e. the span of the membership function A_1, during the backward - pass.

During the first part of a learning period, the input values of one set are treated through all layers of the network, in order to receive one output $O^{(6)}$ as mentioned in the last section. To reduce the error $e = (y - O^{(6)})$ evoked from a, it is necessary to examine the influence this tuning - parameter (a) had on the output (O^6). By means of introducing the squared error E:

$$E = \frac{1}{2}(y - O^{(6)})^2, \tag{3.23}$$

and the the learning - rate η, which is a constant step of iteration, it is possible to compute the improvement Δa of a:

$$\Delta a = -\eta\frac{\partial E}{\partial a} = -\eta\frac{\partial O^{(6)}}{\partial a} \tag{3.24}$$

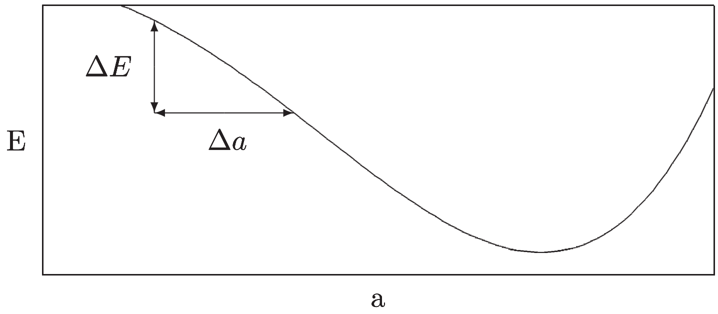

Figure 3.5: Parameter-Tuning

Taking into account a chained relation between a and $O^{(6)}$(see Fig.3.4):

$$O^{(6)}(a) = O^{(6)}(O^{(4)}(O^{(3)}(O^{(2)}(a))))\tag{3.25}$$

containing the following subfunctions:

$$O^{(2)}, O^{(3)} \& \ O^{(4)}\tag{3.26}$$

the improvement Δa is finally obtained as :

$$\begin{aligned}\Delta a &= -\eta \frac{\partial E}{\partial O^{(6)}} \quad \cdot \frac{\partial O^{(6)}}{\partial O^{(4)}} \cdot \frac{\partial O^{(4)}}{\partial O^{(3)}} \quad \cdot \frac{\partial O^{(3)}}{\partial O^{(2)}} \cdot \frac{\partial O^{(2)}}{\partial a} \\ &= -\eta \frac{\partial E}{\partial O^{(6)}} \quad \cdot \frac{\partial O^{(6)}}{\partial \overline{w}_i} \cdot \frac{\partial \overline{w}_i}{\partial w_i} \qquad \cdot \frac{\partial w_i}{\partial \mu_i} \cdot \frac{\partial \mu_i}{\partial a}\end{aligned}\tag{3.27}$$

since by means of the chaine rule:

$$f'(g(h(x))) = f(x) = f'(g) \cdot g'(h) \cdot h'(x)\tag{3.28}$$

3.4 Implementation of ANFIS

Within the large offer of techniques to implement a FIS by means of a NN, the software Matlab/Simulink proposes the possibility of implementing fast, through the use of a special Fuzzy - Toolbox, which includes an environment to build ANFIS'. Moreover it is reasonable to use this software, since the simulation results i.e. the input - output relations are given in a matlab format.

3.4.1 Training - Data selection

In this subsection, the chosen training data set is described.
Before deriving a model, it is important to generate a representative set of training data containing related input - output pairs. Considering the 3 parameters which impact the touchdown velocity deterministically, the set has to exhibit following shape:

$$\begin{bmatrix} \gamma_{fip\ 1} & \gamma_{rwy\ 1} & v_{wlong\ 1} \\ \gamma_{fip\ 2} & \gamma_{rwy\ 2} & v_{wlong\ 2} \\ \vdots & \vdots & \vdots \\ \gamma_{fip\ n} & \gamma_{rwy\ n} & v_{wlong\ n} \end{bmatrix} \Rightarrow \begin{bmatrix} v_{zimp\ 1} \\ v_{zimp\ 2} \\ \vdots \\ v_{zimp\ n} \end{bmatrix} \tag{3.29}$$

It is important, to create multiple parameter constellations, in order to store as much information as possible into the set. Only this way, the model can get representative. Since the training data are obtained from simulation, the key element is the used simulation scenario.
The used scenario consists of "10.000" landings. Within these landings the 3 parameters have been varied through applying a sweep. Each parameter varies within a reasonable, predefined field. The opearating areas of the wind velocity (v_{wlong}) and the runway slope (γ_{rwy}) are derived from the certification (xXXX) - specification for automatic landings, since the standard flare - law is used.
The operating field of the initial flight path angle has been derived from the aircraft performances, since the A/C configuration, mass and thrust affect the smallest selectable fpa (γ_{fip}):
Assuming the thrust (T) to act in the longitudinal axis of the A/C, one can deduct the flight path angle out of the equilibrium of the forces in x-and y direction as:

$$\left.\begin{aligned} \sum = 0 = \frac{\rho}{2} \cdot V^2 \cdot S \cdot c_w + mg\ sin(\gamma) - T \\ \sum = 0 = \frac{\rho}{2} \cdot V^2 \cdot S \cdot c_a - mg\ cos(\gamma) \end{aligned}\right\} tan(\gamma) = \frac{T}{mg} - \frac{c_w}{c_a} \tag{3.30}$$

For Airbus - aircrafts, a glide ratio ($\frac{c_a}{c_w}$) of 10 can be assumed during landings. The steepest fpa is hence reached under a situation with 0 - thrust:

$$\gamma = arctan(\gamma) = arctan(0 - \frac{1}{10}) = -5,71^\circ \sim -6^\circ \tag{3.31}$$

A flat approach, primary moves the touchdown point in the longitudinal axis - without significant impact on the touchdown velocity. The upper margin of the fpa has thus been set to a reasonable value of -2.2°.
As a result, the operating fields of the 3 parameters are:

$$
\begin{aligned}
-0.8\% < & \quad \gamma_{rwy} < & +0.8\% \\
-2.2^\circ < & \quad \gamma_{fip} < & -6^\circ \\
-20kt < & \quad v_{wlong} < & +40kt
\end{aligned}
\tag{3.32}
$$

3.4.2 ANFIS Assembly

In general, there are 3 free parameters, that can be adjusted, when building an ANFIS. The first parameter is the number of membership functions related to one input. This value is a compromise between performance and system complexity, since it augments the number of rules dramatically:

$$n_{rules} = n_{mf,x1} \cdot n_{mf,x2} \cdot \ldots n_{mf,xn} \tag{3.33}$$

The second one is the shape of the membership functions. Principally the shapes of a FIS could differ and exhibit any thinkable form. However, in the AN-FIS, the shape of all membership functions has to be set globally, since the computational effort would raise significantly, when relating individual shapes to each membership function. It is reasonable to employ differentiable functions, because the least squaree - algorithm (lse) conducts a differentiation (see equation 3.27). In literature, there is no heuristic about the use of certain shapes in dedicated situations. That means, that in contrast to a FIS - that is build upon human expert knowledge - , there is no rational argumentation, in the choice of the membership function (in a ANFIS).
The last free parameter is the number of training - epochs. Referring to (Jang) and (Negnewitsky) one can assume the ANFIS performance after the first training - epoch to be a representative value of the performance that can be reached using the chosen configuration of parameters.
In literature it is said, that the finding of the system with the highest performance is an iterative process. Consequently the free parameters have to be varied, until a satisfying performance is reached.
This technique has been conducted, to find a powerful touchdown - velocity estimation. During the iteration, several membership functions have been tested. The best result was found under use of bell - shaped membership functions. Afterwards the system has been trained in order to further reduce the error.
Fig3.6 depicts the performance of the deducted model. To evaluate the model a test - set of 10.000 landings has been generated. Through comparison of the estimated and the true touchdown velocity, it is possible to measure the performance of the FIS.
One can see, that the relation follows a 45°- inclined line. Thus the performance is almost equal over the regarded band. It is further to recognize, that the cloud gets less scattered while following the x-axis in negative direction. This is because the training set contained more hard landings than non - hard landings.

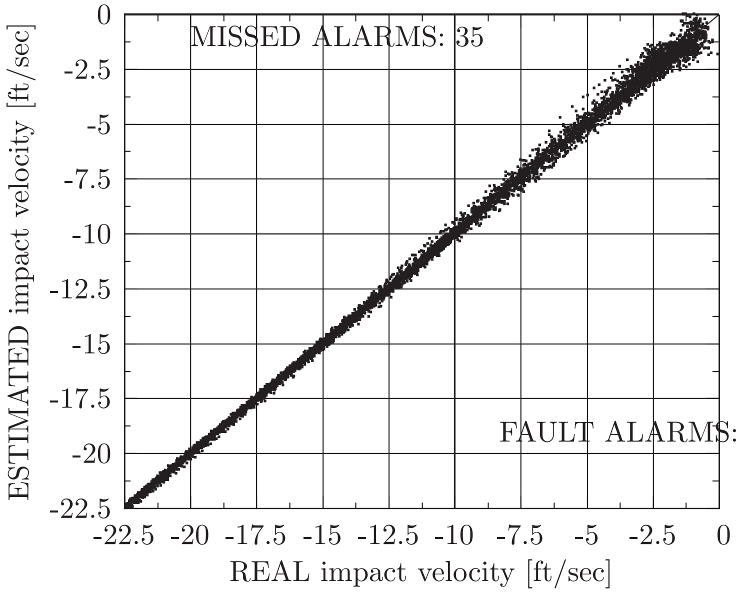

Figure 3.6: comparision between the estimated and the true touchdown velocity

Chapter 4

Turbulences

4.1 Vertical - Wind measurement

In order to estimate the level of vertical turbulences during the approach, the vertical wind has to be observed on - board the A/C. Measured signals which contain information about the wind have to be used therefor. In this context the angle of a attack (α) is a enlightening value, since it is directly dependent from the inflow i.e from the wind direction. A change in the angle of attack, caused by the wind, impacts the flight pass angle γ:

$$\gamma = \Theta - \alpha \tag{4.1}$$

and thus the vertical load

$$n_{z\ tu,c} = \frac{V}{g}\dot{\gamma} + \cos\gamma = \frac{V}{g}(\Theta \dot{-} \alpha) + \cos(\Theta - \alpha) \tag{4.2}$$

Assuming small (fpa) γ during the approach one cane facilitate the equation to:

$$n_{z\ tu,c} = \frac{V}{g}(\Theta \dot{-} \alpha) + 1 \tag{4.3}$$

Calculating the load factor by means of this approach assumes neglecting the A/C inertia i.e. assuming a time constant equal to zero. Indeed, the true load factor is much smaller, because usually the frequency of the vertical wind is high. That means, that the turbulences impact on the load factor is small.
That is why the load factor, as it is given in (4.3) shall be the exaggerated load - factor $n_{zt,cl}$. The true load factor does not explicitely include any information about the vertical wind. In order to obtain the vertical wind velocity, it is however possible to build the difference between the exaggerated, and the true load factor:

$$n_{z\ tu} = n_{z\ tu,c} - n_{z\ tr} \tag{4.4}$$

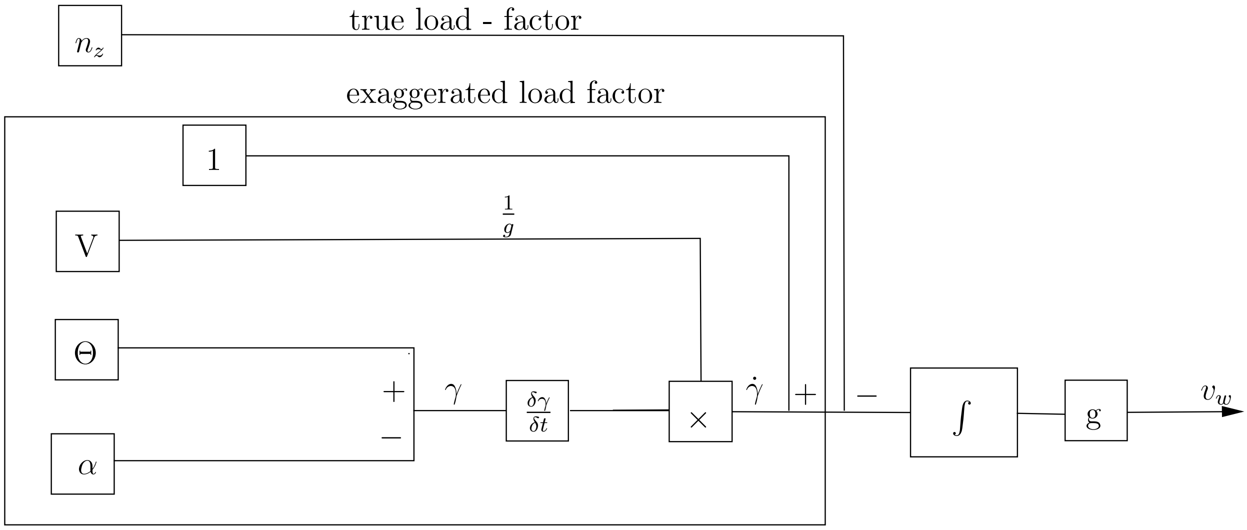

Figure 4.1: turbulence estimator

The integration of the fictive wind load-factor $n_{z\ tu}$ finally directs to the vertical wind velocity: The resulting load-factor is a theoretical value.

Chapter 5

Perspective

Now, that a deterministic calculation and an estimator for the turbulence are at hand, it is finally necessary to combine both algorithms. Therefore I will first have to find a relation between turbulence levels and the diversion in the touchdown velocity (see. Fig.2.6 & Fig.2.7). Afterwards an assembly of both subparts will be done.