2

Air traffic trajectories segmentation based on time-series sensor data

José L. Guerrero, Jesús García and José M. Molina
University Carlos III of Madrid
Spain

1. Introduction

ATC is a critical area related with safety, requiring strict validation in real conditions (Kennedy & Gardner, 1998), being this a domain where the amount of data has gone under an exponential growth due to the increase in the number of passengers and flights. This has led to the need of automation processes in order to help the work of human operators (Wickens et al., 1998). These automation procedures can be basically divided into two different basic processes: the required online tracking of the aircraft (along with the decisions required according to this information) and the offline validation of that tracking process (which is usually separated into two sub-processes, segmentation (Guerrero & Garcia, 2008), covering the division of the initial data into a series of different segments, and reconstruction (Pérez et al., 2006, García et al., 2007), which covers the approximation with different models of the segments the trajectory was divided into). The reconstructed trajectories are used for the analysis and evaluation processes over the online tracking results.

This validation assessment of ATC centers is done with recorded datasets (usually named opportunity traffic), used to reconstruct the necessary reference information. The reconstruction process transforms multi-sensor plots to a common coordinates frame and organizes data in trajectories of an individual aircraft. Then, for each trajectory, segments of different modes of flight (MOF) must be identified, each one corresponding to time intervals in which the aircraft is flying in a different type of motion. These segments are a valuable description of real data, providing information to analyze the behavior of target objects (where uniform motion flight and maneuvers are performed, magnitudes, durations, etc). The performance assessment of ATC multisensor/multitarget trackers require this reconstruction analysis based on available air data, in a domain usually named opportunity trajectory reconstruction (OTR), (Garcia et al., 2009).

OTR consists in a batch process where all the available real data from all available sensors is used in order to obtain smoothed trajectories for all the individual aircrafts in the interest area. It requires accurate original-to-reconstructed trajectory's measurements association, bias estimation and correction to align all sensor measures, and also adaptive multisensor smoothing to obtain the final interpolated trajectory. It should be pointed out that it is an off-line batch processing potentially quite different to the usual real time data fusion systems used for ATC, due to the differences in the data processing order and its specific

processing techniques, along with different availability of information (the whole trajectory can be used by the algorithms in order to perform the best possible reconstruction).
OTR works as a special multisensor fusion system, aiming to estimate target kinematic state, in which we take advantage of both past and future target position reports (smoothing problem). In ATC domain, the typical sensors providing data for reconstruction are the following:

- Radar data, from primary (PSR), secondary (SSR), and Mode S radars (Shipley, 1971). These measurements have random errors in the order of the hundreds of meters (with a value which increases linearly with distance to radar).
- Multilateration data from Wide Area Multilateration (WAM) sensors (Yang et al., 2002). They have much lower errors (in the order of 5-100 m), also showing a linear relation in its value related to the distance to the sensors positions.
- Automatic dependent surveillance (ADS-B) data (Drouilhet et al., 1996). Its quality is dependent on aircraft equipment, with the general trend to adopt GPS/GNSS, having errors in the order of 5-20 meters.

The complementary nature of these sensor techniques allows a number of benefits (high degree of accuracy, extended coverage, systematic errors estimation and correction, etc), and brings new challenges for the fusion process in order to guarantee an improvement with respect to any of those sensor techniques used alone.
After a preprocessing phase to express all measurements in a common reference frame (the stereographic plane used for visualization), the studied trajectories will have measurements with the following attributes: detection time, stereographic projections of its x and y components, covariance matrix, and real motion model (MM), (which is an attribute only included in simulated trajectories, used for algorithm learning and validation). With these input attributes, we will look for a domain transformation that will allow us to classify our samples into a particular motion model with maximum accuracy, according to the model we are applying.
The movement of an aircraft in the ATC domain can be simplified into a series of basic MM's. The most usually considered ones are uniform, accelerated and turn MM's. The general idea of the proposed algorithm in this chapter is to analyze these models individually and exploit the available information in three consecutive different phases.
The first phase will receive the information in the common reference frame and the analyzed model in order to obtain, as its output data, a set of synthesized attributes which will be handled by a learning algorithm in order to obtain the classification for the different trajectories measurements. These synthesized attributes are based on domain transformations according to the analyzed model by means of local information analysis (their value is based on the definition of segments of measurements from the trajectory).They are obtained for each measurement belonging to the trajectory (in fact, this process can be seen as a data pre-processing for the data mining techniques (Famili et al., 1997)).
The second phase applies data mining techniques (Eibe, 2005) over the synthesized attributes from the previous phase, providing as its output an individual classification for each measurement belonging to the analyzed trajectory. This classification identifies the measurement according to the model introduced in the first phase (determining whether it belongs to that model or not).
The third phase, obtaining the data mining classification as its input, refines this classification according to the knowledge of the possible MM's and their transitions,

correcting possible misclassifications, and provides the final classification for each of the trajectory's measurement. This refinement is performed by means of the application of a filter.

Finally, segments are constructed over those classifications (by joining segments with the same classification value). These segments are divided into two different possibilities: those belonging to the analyzed model (which are already a final output of the algorithm) and those which do not belong to it, having to be processed by different models. It must be noted that the number of measurements processed by each model is reduced with each application of this cycle (due to the segments already obtained as a final output) and thus, more detailed models with lower complexity should be applied first. Using the introduced division into three MM's, the proposed order is the following: uniform, accelerated and finally turn model. Figure 1 explains the algorithm's approach:

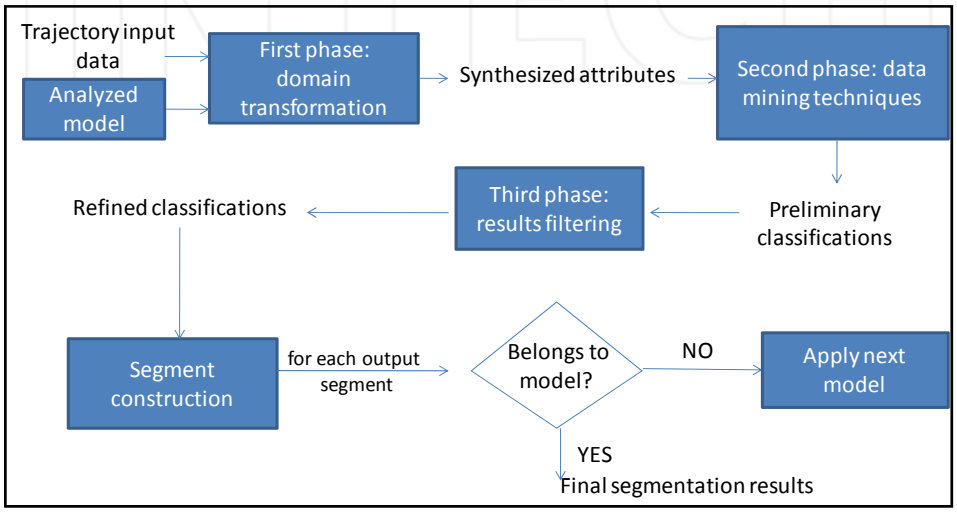

Fig. 1. Overview of the algorithm's approach

The validation of the algorithm is carried out by the generation of a set of test trajectories as representative as possible. This implies not to use exact covariance matrixes, (but estimations of their value), and carefully choosing the shapes of the simulated trajectories. We have based our results on four types of simulated trajectories, each having two different samples. Uniform, turn and accelerated trajectories are a direct validation of our three basic MM's. The fourth trajectory type, racetrack, is a typical situation during landing procedures. The validation is performed, for a fixed model, with the results of its true positives rate (TPR, the rate of measurements correctly classified among all belonging to the model) and false positives rate (FPR, the rate of measurements incorrectly classified among all not belonging the model). This work will show the results of the three consecutive phases using a uniform motion model.

The different sections of this work will be divided with the following organization: the second section will deal with the problem definition, both in general and particularized for the chosen approach. The third section will present in detail the general algorithm, followed

by three sections detailing the three phases for that algorithm when the uniform movement model is applied: the fourth section will present the different alternatives for the domain transformation and choose between them the ones included in the final algorithm, the fifth will present some representative machine learning techniques to be applied to obtain the classification results and the sixth the filtering refinement over the previous results will be introduced, leading to the segment synthesis processes. The seventh section will cover the results obtained over the explained phases, determining the used machine learning technique and providing the segmentation results, both numerically and graphically, to provide the reader with easy validation tools over the presented algorithm. Finally a conclusions section based on the presented results is presented.

2. Problem definition

2.1 General problem definition

As we presented in the introduction section, each analyzed trajectory (T^i) is composed of a collection of sensor reports (or measurements), which are defined by the following vector:

$$\vec{x}_j^i = \left(x_j^i, y_j^i, t_j^i, R_j^i\right), j \in \{1, \dots, N^i\} \quad (1)$$

where j is the measurement number, i the trajectory number, N is the number of measurements in a given trajectory, x_j^i, y_j^i are the stereographic projections of the measurement, t_j^i is the detection time and R_j^i is the covariance matrix (representing the error introduced by the measuring device). From this problem definition our objective is to divide our trajectory into a series of segments (B_k^i), according to our estimated MOF. This is performed as an off-line processing (meaning that we may use past and future information from our trajectory). The segmentation problem can be formalized using the following notation:

$$T^i = \cup B_k^i \qquad B_k^i = \{x_j^i\} \qquad j \in \{k_{min}, \dots, k_{max}\} \quad (2)$$

In the general definition of this problem these segments are obtained by the comparison with a test model applied over different windows (aggregations) of measurements coming from our trajectory, in order to obtain a fitness value, deciding finally the segmentation operation as a function of that fitness value (Mann et al. 2002), (Garcia et al., 2006).

We may consider the division of offline segmentation algorithms into different approaches: a possible approach is to consider the whole data from the trajectory and the segments obtained as the problem's basic division unit (using a global approach), where the basic operation of the segmentation algorithm is the division of the trajectory into those segments (examples of this approach are the bottom-up and top-down families (Keogh et al., 2003)). In the ATC domain, there have been approaches based on a direct adaptation of online techniques, basically combining the results of forward application of the algorithm (the pure online technique) with its backward application (applying the online technique reversely to the time series according to the measurements detection time) (Garcia et al., 2006). An alternative can be based on the consideration of obtaining a different classification value for each of the trajectory's measurements (along with their local information) and obtaining the

segments as a synthesized solution, built upon that classification (basically, by joining those adjacent measures sharing the same MM into a common segment). This approach allows the application of several refinements over the classification results before the final synthesis is performed, and thus is the one explored in the presented solution in this chapter.

2.2 Local approach problem definition

We have presented our problem as an offline processing, meaning that we may use information both from our past and our future. Introducing this fact into our local representation, we will restrict that information to a certain local segment around the measurement which we would like to classify. These intervals are centered on that measurement, but the boundaries for them can be expressed either in number of measurements, (3), or according to their detection time values (4).

$$B(x_m^i) = \{x_j^i\} \quad j \in [m-p, \dots, m, \dots, m+p] \tag{3}$$

$$B(x_m^i) = \{x_j^i\} \quad t_j^i \epsilon \{t_m^i - \mathrm{n}, \dots, \mathrm{t_m}, \dots, t_m^i + \mathrm{n}\} \tag{4}$$

Once we have chosen a window around our current measurement, we will have to apply a function to that segment in order to obtain its transformed value. This general classification function $F(\overrightarrow{x_j^i})$, using measurement boundaries, may be represented with the following formulation:

$$F(\overrightarrow{x_m^i}) = F(\overrightarrow{x_m^i}|T^i) \Rightarrow F(\overrightarrow{x_j^i}|B(x_m^i)) = F_p(\vec{x}_{m-p}^i, .., \vec{x}_m^i, .., \vec{x}_{m+p}^i) \tag{5}$$

From this formulation of the problem we can already see some of the choices available: how to choose the segments (according to (3) or (4)), which classification function to apply in (5) and how to perform the final segment synthesis. Figure 2 shows an example of the local approach for trajectory segmentation.

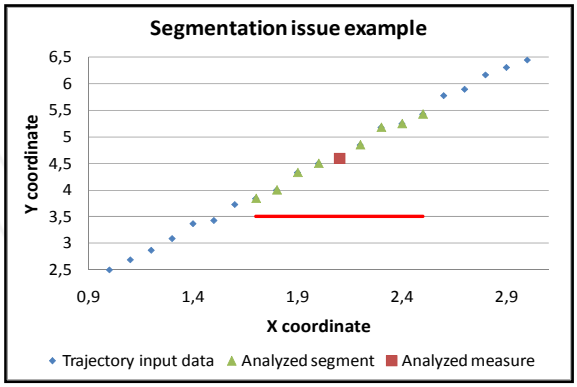

Fig. 2. Local approach for trajectory segmentation approach overview

3. General algorithm proposal

As presented in the introduction section, we will consider three basic MM's and classify our measurements individually according to them (Guerrero & Garcia, 2008). If a measurement is classified as unknown, it will be included in the input data for the next model's analysis. This general algorithm introduces a design criterion based on the introduced concepts of TPR and FPR, respectively equivalent to the type I and type II errors (Allchin, 2001). The design criterion will be to keep a FPR as low as possible, understanding that those measurements already assigned to a wrong model will not be analyzed by the following ones (and thus will remain wrongly classified, leading to a poorer trajectory reconstruction). The proposed order for this analysis of the MM's is the same in which they have been introduced, and the choice is based on how accurately we can represent each of them.

In the local approach problem definition section, the segmentation problem was divided into two different sub-problems: the definition of the $F_p(\overrightarrow{x_m^i})$ function (to perform measurement classification) and a final segment synthesis over that classification.

According to the different phases presented in the introduction section, we will divide the definition of the classification function $F(\overrightarrow{x_j^i})$ into two different tasks: a domain transformation $Dt(\overrightarrow{x_j^i})$ (domain specific, which defines the first phase of our algorithm) and a final classification $Cl(Dt(\overrightarrow{x_j^i}))$ (based on general classification algorithms, represented by the data mining techniques which are introduced in the second phase). The final synthesis over the classification results includes the refinement over that classification introduced by the filtering process and the actual construction of the output segment (third phase of the proposed algorithm).

The introduction of the domain transformation $Dt(\overrightarrow{x_j^i})$ from the initial data in the common reference frame must deal with the following issues: segmentation, (which will cover the decision of using an independent classification for each measurement or to treat segments as an indivisible unit), definition for the boundaries of the segments, which involves segment extension (which analyzes the definition of the segments by number of points or according to their detection time values) and segment resolution (dealing with the choice of the length of those segments, and how it affects our results), domain transformations (the different possible models used in order to obtain an accurate classification in the following phases), and threshold choosing technique (obtaining a value for a threshold in order to pre-classify the measurements in the transformed domain).

The second phase introduces a set of machine learning techniques to try to determine whether each of the measurements belongs to the analyzed model or not, based on the pre-classifications obtained in the first phase. In this second phase we will have to choose a $Cl(Dt(\overrightarrow{x_j^i}))$ technique, along with its configuration parameters, to be included in the algorithm proposal. The considered techniques are decision trees (C4.5, (Quinlan, 1993)) clustering (EM, (Dellaert, 2002)) neural networks (multilayer perceptron, (Gurney, 1997)) and Bayesian nets (Jensen & Graven-Nielsen, 2007) (along with the simplified naive Bayes approach (Rish, 2001)).

Finally, the third phase (segment synthesis) will propose a filter, based on domain knowledge, to reanalyze the trajectory classification results and correct those values which may not follow this knowledge (essentially, based on the required smoothness in MM's

changes). To obtain the final output for the model analysis, the isolated measurements will be joined according to their classification in the final segments of the algorithm.
The formalization of these phases and the subsequent changes performed to the data is presented in the following vectors, representing the input and output data for our three processes:

Input data: $T^i = \left\{\overrightarrow{x_j^i}\right\},\ j\epsilon\{1..N^i\} \quad \vec{x}_j^i = \left(x_j^i, y_j^i, t_j^i, R_j^i\right)$
Domain transformation: $\mathrm{Dt}(\overrightarrow{x_j^i}) \Longrightarrow \mathrm{F}(\overrightarrow{x_j^i}|T^i) \Rightarrow \mathrm{F}(\overrightarrow{x_j^i}|B_j^i) = \{\mathrm{Pc}_k^j\},\ k \in \{1..M\}$
Pc_k^j = pre-classification k for measurement j, M = number of pre-classifications included
Classification process: $\mathrm{Cl}(\mathrm{Dt}(\overrightarrow{x_j^i})) = \mathrm{Cl}(\{\mathrm{Pc}_k^j\}) = C_j$
C_j= automatic classification result for measurement j (including filtering refinement)
Final output: $T^i = \cup B_k^i \quad B_k^i = \{x_j^i\}\ j \in [k_{min}, k_{max}]$
B_k^i= Final segments obtained by the union process

4. Domain transformation

The first phase of our algorithm covers the process where we must synthesize an attribute from our input data to represent each of the trajectory's measurements in a transformed domain and choose the appropriate thresholds in that domain to effectively differentiate those which belong to our model from those which do not do so.
The following aspects are the key parameters for this phase, presented along with the different alternatives compared for them, (it must be noted that the possibilities compared here are not the only possible ones, but representative examples of different possible approaches):

- ***Transformation function: c***orrelation coefficient / Best linear unbiased estimator residue
- ***Segmentation granularity:*** segment study / independent study
- ***Segment extension,*** time / samples, and ***segment resolution,*** length of the segment, using the boundary units imposed by the previous decision
- ***Threshold choosing technique,*** choice of a threshold to classify data in the transformed domain.

Each of these parameters requires an individual validation in order to build the actual final algorithm tested in the experimental section. Each of them will be analyzed in an individual section in order to achieve this task.

4.1 Transformation function analysis

The transformation function decision is probably the most crucial one involving this first phase of our algorithm. The comparison presented tries to determine whether there is a real accuracy increase by introducing noise information (in the form of covariance matrixes). This section compares a correlation coefficient (Meyer, 1970) (a general statistic with no noise information) with a BLUE residue (Kay, 1993) (which introduces the noise in the measuring process). This analysis was originally proposed in (Guerrero & Garcia, 2008). The equations for the CC statistical are the following:

$$CC(x_j|B_k) = \frac{\sigma_{xy}(x_j|B_k)}{\sigma_x(x_j|B_k)\sigma_y(x_j|B_k)} \quad \sigma_x(x_j|B_k) = \sqrt{\frac{\sum_{i=\text{kmin}}^{\text{kmax}}(x_i-\bar{x})^2}{\text{kmax}-\text{kmin}}}$$
$$\sigma_{xy}(x_j|B_k) = \frac{1}{\text{kmax}-\text{kmin}+1}\sum_{i=\text{kmin}}^{kmax}(x_i-\bar{x})(y_i-\bar{y}) \tag{6}$$

In order to use the BLUE residue we need to present a model for the uniform MM, represented in the following equations:

$$\vec{x}_m(k) = \begin{bmatrix} x_m(k) \\ y_m(k) \end{bmatrix} = \begin{bmatrix} 1 & t_k & 0 & 0 \\ 0 & 0 & 1 & t_k \end{bmatrix} \begin{bmatrix} x_0 \\ vx_0 \\ y_0 \\ vy_0 \end{bmatrix} + \begin{bmatrix} n_x(k) \\ n_y(k) \end{bmatrix} = H(t_k)\vec{\theta} + \vec{n}(k) \tag{7}$$

$$\langle\vec{\theta}\rangle = \begin{bmatrix} \langle x_0\rangle \\ \langle vx_0\rangle \\ \langle y_0\rangle \\ \langle vy_0\rangle \end{bmatrix} = \left(\sum_k H(t_k)^T R_k^{-1} H(t_k)\right)^{-1} \sum_k H(t_k)^T R_k^{-1} \overrightarrow{x_m}(k) \tag{8}$$

With those values we may calculate the interpolated positions for our two variables and the associated residue:

$$x_{int}(t) = \langle x_0\rangle + \langle vx_0\rangle t \qquad y_{int}(t) = \langle y_0\rangle + \langle vy_0\rangle t \tag{9}$$

$$res(x_j|B_k) = \frac{1}{(kmax-kmin+1)}\sum_{k=kmin}^{k=kmax}\begin{pmatrix} x(k)-x_{int}(k) & y(k)-y_{int}(k) \end{pmatrix} R_k^{-1} \begin{pmatrix} x(k)-x_{int}(k) \\ y(k)-y_{int}(k) \end{pmatrix} \tag{10}$$

The BLUE residue is presented normalized (the residue divided by the length of the segment in number of measurements), in order to be able to take advantage of its interesting statistical properties, which may be used into the algorithm design, and hence allow us to obtain more accurate results if it is used as our transformation function.

To obtain a classification value from either the CC or the BLUE residue value these values must be compared with a certain threshold. The CC threshold must be a value close, in absolute value, to 1, since that indicates a strong correlation between the variables. The BLUE residue threshold must consider the approximation to a chi-squared function which can be performed over its value (detailed in the threshold choosing technique section). In any case, to compare their results and choose the best technique between them, the threshold can be chosen by means of their TPR and FPR values (choosing manually a threshold which has zero FPR value with the highest possible TPR value).

To facilitate the performance comparison between the two introduced domain transformations, we may resort to ROC curves (Fawcett, 2006), which allow us to compare their behavior by representing their TPR against their FPR. The result of this comparison is shown in figure 3.

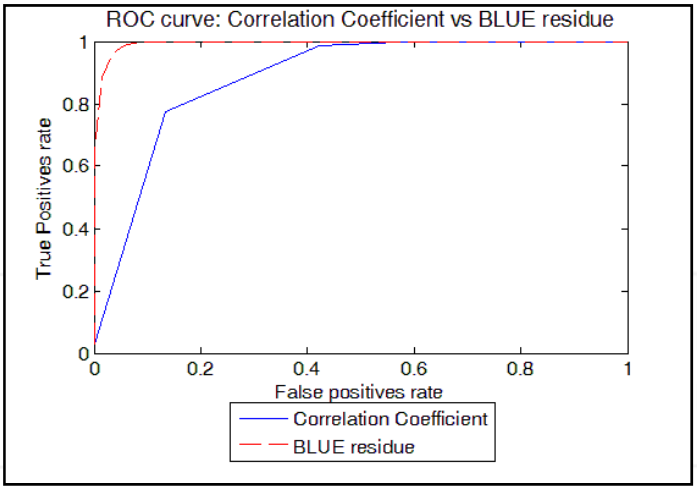

Fig. 3. Comparison between the two presented domain transformations: CC and BLUE residue

The comparison result shows that the introduction of the sensor's noise information is vital for the accuracy of the domain transformation, and thus the BLUE residue is chosen for this task.

4.2 Segmentation granularity analysis

Having chosen the BLUE residue as the domain transformation function, we intend to compare the results obtained with two different approaches, regarding the granularity they apply: the first approach will divide the trajectory into a series of segments of a given size (which may be expressed, as has been presented, in number of measurements of with detection time boundaries), obtain their synthesized value and apply that same value to every measurement belonging to the given segment. On the other hand, we will use the approach presented in the local definition of the problem, that, for every measurement belonging to the trajectory, involves choosing a segment around the given measurement, obtain its surrounding segment and find its transformed value according to that segment (which is applied only to the central measurement of the segment, not to every point belonging to it).

There are a number of considerations regarding this comparison: obviously, the results achieved by the local approach obtaining a different transformed value for each measurement will be more precise than those obtained by its alternative, but it will also involve a greater computational complexity. Considering a segment size of s_size and a trajectory with n measurements, the complexity of obtaining a transformed value for each of these measurements is $O(n * s\_size)$ whereas obtaining only a value and applying it to the whole segment is $O(n)$, introducing efficiency factors which we will ignore due to the offline nature of the algorithm.

Another related issue is the restrictions which applying the same transformed value to the whole segment introduces regarding the choice of those segments boundaries. If the transformed value is applied only to the central measurement, we may choose longer of

shorter segments according to the transformation results (this choice will be analysed in the following section), while applying that same transformed value to the whole segments introduces restrictions related to the precision which that length introduces (longer segments may be better to deal with the noise in the measurements, but, at the same time, obtain worse results due to applying the same transformed value to a greater number of measurements).
The ROC curve results for this comparison, using segments composed of thirty-one measurements, are shown in figure 4.

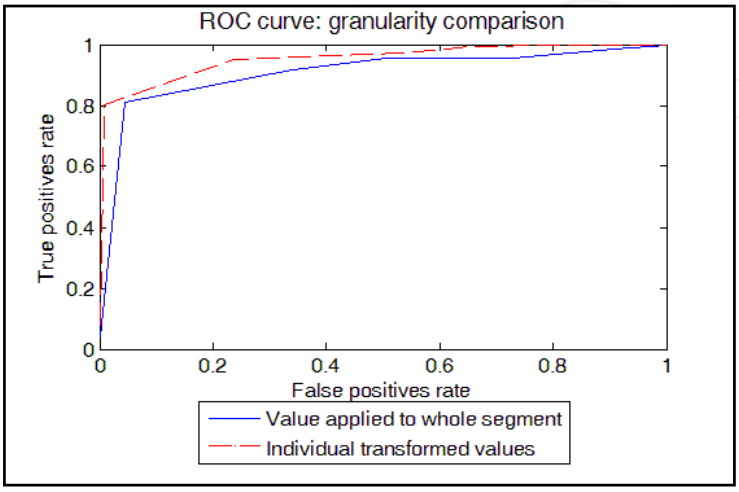

Fig. 4. Comparison between the two presented granularity choices

Given the presented design criterion, which remarks the importance of low FPR values, we may see that individual transformed values perform much better at that range (leftmost side of the figure), leading us, along with the considerations previously exposed, to its choice for the algorithm final implementation.

4.3 Segment definition analysis

The definition of the segments we will analyze involves two different factors: the boundary units used and the length (and its effects on the results) of those segments (respectively referred to as segment extension and segment resolution in this phase's presentation). One of the advantages of building domain-dependent algorithms is the use of information belonging to that domain. In the particular case of the ATC domain, we will have information regarding the lengths of the different possible manoeuvres performed by the aircrafts, and will base our segments in those lengths. This information will usually come in the form of time intervals (for example, the maximum and minimum duration of turn manoeuvres in seconds), but may also come in the form on number of detections in a given zone of interest. Thus, the choice of one or the other (respectively represented in the problem definition section by equations (4) and (3)) will be based on the available information.

With the units given by the available information, Figure 5 shows the effect of different resolutions over a given turn trajectory, along with the results over those resolutions.

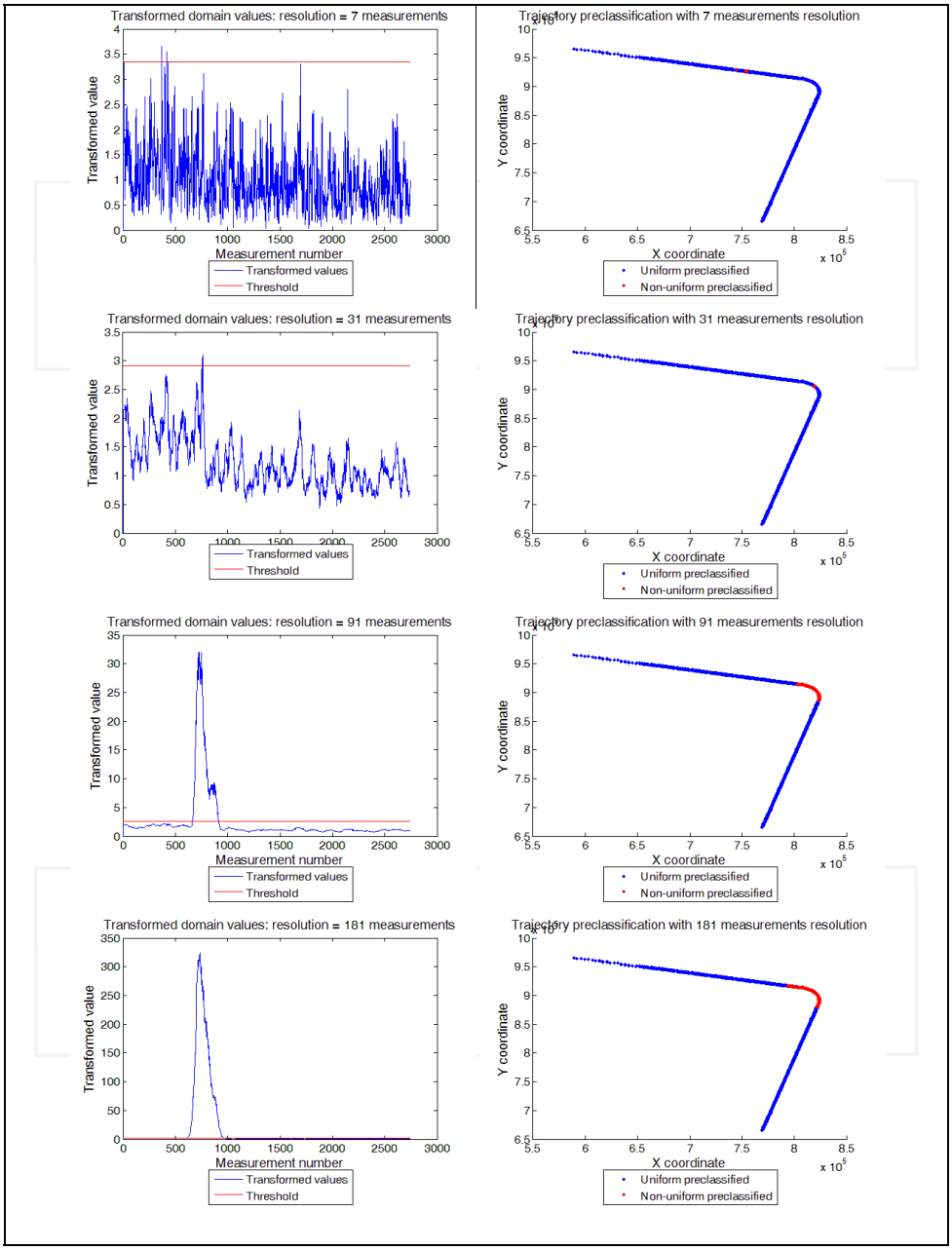

Fig. 5. Comparison of transformed domain values and pre-classification results

Observing the presented results, where the threshold has been calculated according to the procedure explained in the following section, we may determine the resolution effects: short segments exhibit several handicaps: on the one hand, they are more susceptible to the noise effects, and, on the other hand, in some cases, long smooth non-uniform MM segments may be accurately approximated with short uniform segments, causing the algorithm to bypass them (these effects can be seen in the lower resolutions shown in figure 5). Longer segments allow us to treat the noise effects more effectively (with resolution 31 there are already no misclassified measurements during non-uniform segments) and make the identification of non-uniform segments possible, avoiding the possibility of obtaining an accurate approximation of these segments using uniform ones (as can be seen with resolution 91) However, long segments also make the measurements close to a non-uniform MM increase their transformed value (as their surrounding segment starts to get into the non-uniform MM), leading to the fact that more measurements around the non-uniform segments will be pre-classified incorrectly as non-uniform (resolution 181). A different example of the effects of resolution in these pre-classification results may be looked up in (Guerrero et al., 2010).
There is, as we have seen, no clear choice for a single resolution value. Lower resolutions may allow us to obtain more precise results at the beginning and end of non-uniform segments, while higher resolution values are capital to guarantee the detection of those non-uniform segments and the appropriate treatment of the measurements noise. Thus, for this first phase, a multi-resolution approach will be used, feeding the second phase with the different pre-classifications of the algorithm according to different resolution values.

4.4 Threshold choosing technique

The threshold choice involves automatically determining the boundary above which transformed measurements will be considered as unknown. Examples of this choice may be seen in the previous section (figure 5). According to our design criterion, we would like to obtain a TPR as high as possible keeping our FPR ideally at a zero value. Graphically over the examples in figure 5 (especially for the highest resolutions, where the non-uniform maneuver can be clearly identified), that implies getting the red line as low as possible, leaving only the central section over it (where the maneuver takes place, making its residue value high enough to get over our threshold).
As presented in (Guerrero et al., 2010), the residue value in (10) follows a Chi-squared probability distribution function (pdf) normalized by its degrees of freedom, n. The value of n is given by twice the number of 2D measurements contained in the interval minus the dimension of P (P=4 in the presented uniform model, as we are imposing 4 linear restrictions). For a valid segment residual, "res" behaves with distribution $\frac{1}{(kmax-kmin+1)}\chi^2_{2(kmax-kmin+1)-P}$, which has the following mean and variance:

$$\mu = 2 - \frac{P}{(kmax-kmin+1)} \qquad \sigma^2 = \frac{4}{(kmax-kmin+1)} - \frac{2P}{(kmax-kmin+1)^2} \tag{11}$$

The residue distribution allows us to establish our criterion based on the TPR value, but not the FPR (we have a distribution over the uniform measurements, not the unknown ones), which is the one constrained by the design criterion. We may use the Chevychev's inequality (Meyer, 1970) to determine a threshold which should leave the 99% of the measurements belonging to our model above it (TPR>=0.99), with $\mu + 3\sigma$ value. From the values exposed in (11) we get the following threshold value:

$$\text{thres}=2-\frac{4}{N}+3\sqrt{\frac{4}{N}-\frac{8}{N^2}} \quad N=(kmax-kmin+1) \tag{12}$$

This threshold depends on the resolution of the segment, *N*, which also influences the residue value in (10). It is interesting to notice that the highest threshold value is reached with the lowest resolution. This is a logical result, since to be able to maintain the TPR value (having fixed it with the inequality at 99%) with short segments, a high threshold value is required, in order to counteract the noise effects (while longer segments are more resistant to that noise and thus the threshold value may be lower).

We would like to determine how precisely our χ^2 distribution represents our normalized residue in non-uniform trajectories with estimated covariance matrix. In the following figures we compare the optimal result of the threshold choice (dotted lines), manually chosen, to the results obtained with equation (12). Figure 6 shows the used trajectories for this comparison, along with the proposed comparison between the optimal TPR and the one obtained with (12) for increasing threshold values.

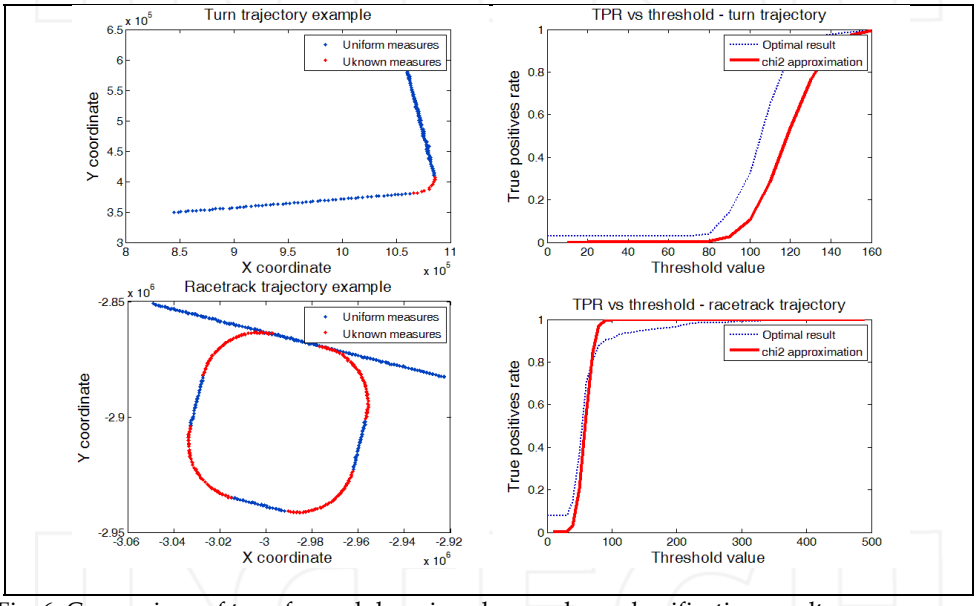

Fig. 6. Comparison of transformed domain values and pre-classification results

In the two trajectories in figure 6 we may appreciate two different distortion effects introduced by our approximation. The turn trajectory shows an underestimation of our TPR due to the inexactitude in the covariance matrix R_k. This inexactitude assumes a higher noise than the one which is present in the trajectory, and thus will make us choose a higher threshold than necessary in order to obtain the desired TPR margin.

In the racetrack trajectory we perceive the same underestimation at the lower values of the threshold, but then our approximation crosses the optimal results and reaches a value over it. This is caused by the second distortion effect, the maneuver's edge measurements. The measurements close to a maneuver beginning or end tend to have a higher residue value

than the theoretical one for a uniform trajectory (due to their proximity to the non-uniform segments), making us increase the threshold value to classify them correctly (which causes the optimal result to show a lower TPR in the figure). These two effects show that a heuristic tuning may be required in our χ^2 distribution in order to adapt it to these distortion effects.

5. Machine learning techniques application

The algorithm's first phase, as has been detailed, ended with a set of pre-classification values based on the application of the domain transformation with different resolutions to every measurement in the trajectory. The objective of this second phase is to obtain a classification according to the analyzed model for each of these measurements, to be able to build the resulting segments from this data.

There are countless variants of machine learning techniques, so the choice of the ones presented here was not a trivial one. There was not a particular family of them more promising a-priori, so the decision tried to cover several objectives: they should be easy to replicate, general and, at the same time, cover different approaches in order to give the algorithm the chance to include the best alternative from a wide set of choices. This led to the choice of Weka®[1] as the integrated tool for these tests, trying to use the algorithms with their default parameters whenever possible (it will be indicated otherwise if necessary), even though the fine tuning of them gives us a very slight better performance, and the choice of representative well tested algorithms from different important families in machine learning: decision trees (C4.5) clustering (EM) neural networks (multilayer perceptron) and Bayesian networks, along with the simplified naive Bayes approach. We will describe each of these techniques briefly.

Decision trees are predictive models based on a set of "attribute-value" pairs and the entropy heuristic. The C 4.5 algorithm (Quinlan, 1993) allows continuous values for its variables.

Clustering techniques have the objective of grouping together examples with similar characteristics and obtain a series of models for them that, even though they may not cover all the characteristics of their represented members, can be representative enough of their sets as a whole (this definition adapts very well to the case in this chapter, since we want to obtain a representative set of common characteristics for measurements following our analyzed model). The EM algorithm (Dellaert, 2002) is based on a statistical model which represents the input data basing itself on the existence of *k* Gaussian probability distribution functions, each of them representing a different cluster. These functions are based on maximum likelihood hypothesis. It is important to realize that this is an unsupervised technique which does not classify our data, only groups it. In our problem, we will have to select the classification label afterwards for each cluster. In this algorithm, as well, we will introduce a non standard parameter for the number of clusters. The default configuration allows Weka to automatically determine this number, but, in our case, we only want two different clusters: one representing those measurements following the analyzed model and a different one for those unknown, so we will introduce this fact in the algorithm's configuration.

[1] Available online at http://www.cs.waikato.ac.nz/ml/weka/

Bayesian networks (Jensen & Graven-Nielsen, 2007) are directed acyclic graphs whose nodes represent variables, and whose missing edges encode conditional independencies between the variables. Nodes can represent any kind of variable, be it a measured parameter, a latent variable or a hypothesis. Special simplifications of these networks are Naive Bayes networks (Rish, 2001), where the variables are considered independent. This supposition, even though it may be considered a very strong one, usually introduces a faster learning when the number of training samples is low, and in practice achieves very good results.
Artificial neural networks are computational models based on biological neural networks, consisting of an interconnected group of artificial neurons, which process information using a connectionist approach to computation. Multilayer Perceptron (MP), (Gurney, 1997), are feed-forward neural networks having an input layer, an undetermined number of hidden layers and an output layer, with nonlinear activation functions. MP's are universal function approximators, and thus they are able to distinguish non-linearly separable data. One of the handicaps of their approach is the configuration difficulties which they exhibit (dealing mainly with the number of neurons and hidden layers required for the given problem). The Weka tool is able to determine these values automatically.

6. Classification refinement and segment construction

The algorithm's final phase must refine the results from the machine learning techniques and build the appropriate segments from the individual measurements classification. To perform this refinement, we will use the continuity in the movement of the aircrafts, meaning that no abrupt MM changes can be performed (every MM has to be sustained for a certain time-length). This means that situations where a certain measurement shows a classification value different to its surrounding values can be corrected assigning to it the one shared by its neighbours.
This correction will be performed systematically by means of a voting system, assigning the most repeated classification in its segment to the central measurement. This processing is similar to the one performed by median filters (Yin et al., 1996) widely used in image processing (Baxes, 1994).
The widow size for this voting system has to be determined. In the segment definition section the importance of the available information regarding the length of the possible non-uniform MM's was pointed out, in order to determine the resolution of the domain transformation, which is used as well for this window size definition. Choosing a too high value for our window size might cause the algorithm to incorrectly reclassify non-uniform measurements as uniform (if its value exceeds the length of the non-uniform segment they belong to) leading to an important increase in the FPR value (while the design criterion tries to avoid this fact during the three phases presented). Thus, the window size will have the value of the shortest possible non-uniform MM.
It also important to determine which measurements must be treated with this filtering process. Through the different previous phases the avoidance of FPR has been highlighted (by means of multi-resolution domain transformation and the proper election of the used machine learning technique), even at the cost of slightly decreasing the TPR value. Those considerations are followed in this final phase by the application of this filtering process only to measurements classified as non-uniform, due to their possible misclassification

caused by their surrounding noise. Figure 7 shows the results of this filtering process applied to an accelerated trajectory

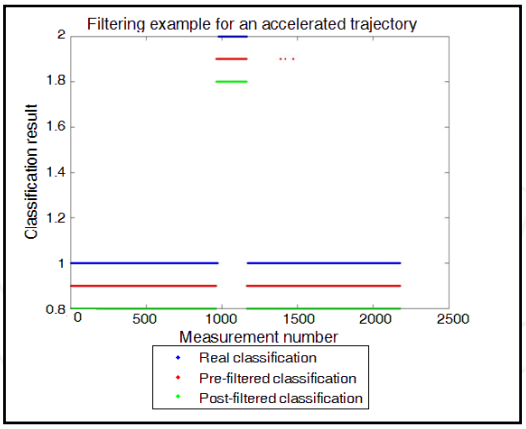

Fig. 7. Example filtering process applied to an accelerated trajectory

In figure 7, the lowest values (0.8 for post-filtered results, 0.9 for pre-filtered ones and 1 for the real classification) indicate that the measurement is classified as uniform, whereas their respective higher ones (1+ its lowest value) indicate that the measurement is classified as non-uniform. This figure shows that some measurements previously misclassified as non-uniform are corrected.

The importance of this filtering phase is not usually reflected in the TPR, bearing in mind that the number of measurements affected by it may be very small, but the number of output segments can vary its value significantly. In the example in figure 7, the pre-filtered classification would have output nine different segments, whereas the post-filtered classification outputs only three segments. This change highlights the importance of this filtering process.

The method to obtain the output segments is extremely simple after this median filter application: starting from the first detected measurement, one segment is built according to that measurement classification, until another measurement *i* with a different classification value is found. At that point, the first segment is defined with boundaries *[1, i-1]* and the process is restarted at measurement *i*, repeating this cycle until the end of the trajectory is reached.

7. Experimental validation

The division of the algorithm into different consecutive phases introduces validation difficulties, as the results are mutually dependant. In this whole work, we have tried to show those validations along with the techniques explanation when it was unavoidable (as occurred in the first phase, due to the influence of the choices in its different parameters) and postpone the rest of the cases for a final validation over a well defined test set (second and third phases, along with the overall algorithm performance).

This validation process is carried out by the generation of a set of test trajectories as representative as possible, implying not to use exact covariance matrixes, (but estimations of their value), and carefully choosing the shapes of the simulated trajectories. We have based our results on four kind of simulated trajectories, each having two different samples. Uniform, turn and accelerated trajectories are a direct validation of our three basic MM's identified, while the fourth trajectory type, racetrack, is a typical situation during landing procedures.

This validation will be divided into three different steps: the first one will use the whole data from these trajectories, obtain the transformed multi-resolution values for each measurement and apply the different presented machine learning techniques, analyzing the obtained results and choosing a particular technique to be included in the algorithm as a consequence of those results.

Having determined the used technique, the second step will apply the described refinement process to those classifications, obtaining the final classification results (along with their TPR and FPR values). Finally the segmentations obtained for each trajectory are shown along with the real classification of each trajectory, to allow the reader to perform a graphical validation of the final results.

7.1 Machine learning techniques validation

The validation method for the machine learning techniques still has to be determined. The chosen method is cross-validation (Picard and Cook, 1984) with 10 folds. This method ensures robustness in the percentages shown. The results output format for any of these techniques in Weka provides us with the number of correctly and incorrectly classified measurements, along with the confusion matrix, detailing the different class assignations. In order to use these values into our algorithm's framework, they have to be transformed into TPR and FPR values. They can be obtained from the confusion matrix, as shown in the following example:

Weka's raw output:

Correctly Classified Instances 10619 96.03 %
Incorrectly Classified Instances 439 3.97 %
=== Confusion Matrix ===

```
  a   b   <-- classified as
345  37 |   a = uniform_model
  0 270 |   b = unknown_model
```

Algorithm parameters:

TPR = 345/37 = 0,903141361 FPR = 0/270 = 0

The selection criterion from these values must consider the design criterion of keeping a FPR value as low as possible, trying to obtain, at the same time, the highest possible TPR value. Also, we have introduced as their input only six transformed values for each measurement, corresponding to resolutions 11, 31, 51, 71, 91 and 111 (all of them expressed in number of measurements) The results presentation shown in table 1 provides the individual results for each trajectory, along with the results when the whole dataset is used as its input. The individual results do not include the completely uniform trajectories (due to their lack of FPR, having no non-uniform measurements). Figure 8 shows the graphical comparison of the different algorithms with the whole dataset according to their TPR and FPR values

Trajectory	C 4.5		EM Clustering		Bayesian networks		Naive Bayes		Multilayer perceptron	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
Racetr. 1	0,903	0	0,719	0	0,903	0	0,903	0	0,903	0
Racetr. 2	0,966	0,036	0,625	0	0,759	0	0,759	0	0,966	0,036
Turn 1	0,975	0	1	1	0,918	0	0,914	0	0,975	0
Turn 2	0,994	0,019	0,979	0	0,987	0	0,987	0	0,994	0,019
Accel. 1	0,993	0	0,993	0	0,993	0	0,993	0	0,993	0
Accel. 2	0,993	0,021	0,993	0,021	0,993	0,021	0,993	0,021	0,993	0,021
Whole dataset	0,965	0,078	0,941	0,003	0,956	0,096	0,956	0,096	0,973	0,155

Table 1. Results presentation over the introduced dataset for the different proposed machine learning techniques

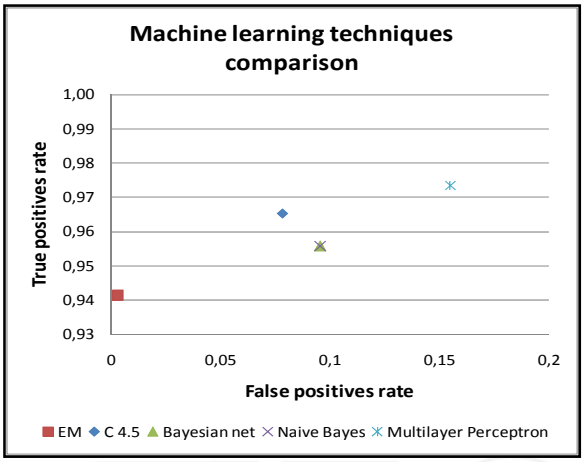

Fig. 8. TPR and FPR results comparison for the different machine learning techniques over the whole dataset.

From the results above we can determine that the previous phase has performed an accurate job, due to the fact that all the different techniques are able to obtain high TPR and low TPR results. When we compare them, the relationship between the TPR and the FPR does not allow a clear choice between the five techniques. If we recur to multi-objetive optimization terminology (Coello et al. 2007), (which is, in fact, what we are performing, trying to obtain a FPR as low as possible with a TPR as high as possible) we may discard the two Bayesian approaches, as they are dominated (in terms of Pareto dominance) by the C 4.5 solution. That leaves us the choice between EM (with the lowest FPR value), the C 4.5 (the most equilibrated between FPR and TPR values) and the multilayer perceptron (with the highest TPR). According to our design criterion, we will incorporate into the algorithm the technique with the lowest FPR: EM clustering.

7.2 Classification refinement validation

To obtain a more detailed performance analysis over the filtering results, we will detail the TPR and FPR values for each individual trajectory before and after this filtering phase. Also, to obtain a numerical validation over the segmentation quality we will detail the real and output number of segments for each of these trajectories. These results are shown in table 2.

Trajectory	Pre-filtered results		Post-filtered results		Number of segments	
	TPCP	TPFP	TPCP	TPFP	Real	Output
Racetr. 1	0,4686	0	0,4686	0	9	3
Racetr.2	0,5154	0	0,5154	0	9	3
Uniform 1	0,9906	0	1	0	1	1
Uniform 2	0,9864	0	0,9961	0	1	3
Turn 1	0,9909	0,0206	0,994	0,0206	3	3
Turn 2	0,9928	0	0,9942	0	3	3
Accel. 1	0,6805	0	0,6805	0	3	3
Accel. 2	0,9791	0	0,9799	0	3	3

Table 2. Comparison of TPR and FPR values for the dataset's trajectories, along with the final number of segments for this phase

In the previous results we can see that the filtering does improve the results in some trajectories, even though the numerical results over TPR and FPR are not greatly varied (the effect, as commented in the filtering section, is more noticeable in the number of segments, given that every measurement misclassified might have meant the creation of an additional output segment).

The overall segmentation output shows difficulties dealing with the racetrack trajectories. This is caused by the fact that their uniform segments inside the oval are close to two different non-uniform ones, thus increasing their transformed value to typical non-uniform measurements ones, being accordingly classified by the machine learning technique. However, these difficulties decrease the value of TPR, meaning that this misclassification can be corrected by the non-uniform models cycles which are applied after the described uniform one detailed through this work. The rest of the trajectories are segmented in a satisfactory way (all of them show the right number of output segments, apart from an additional non-uniform segment in one of the completely uniform ones, caused by the very high measuring noise in that area).

7.3 Overall graphical validation

Even though the previous section showed the different numerical results for every trajectory, the authors considered that a final visual validation is capital to enable the reader to perform the analysis of the segmentation quality, at least for one example of each kind of the different trajectories (focusing on the difficult cases detailed in the previous section).

Figure 9 shows the original trajectory with its correct classification along with the algorithm's results.

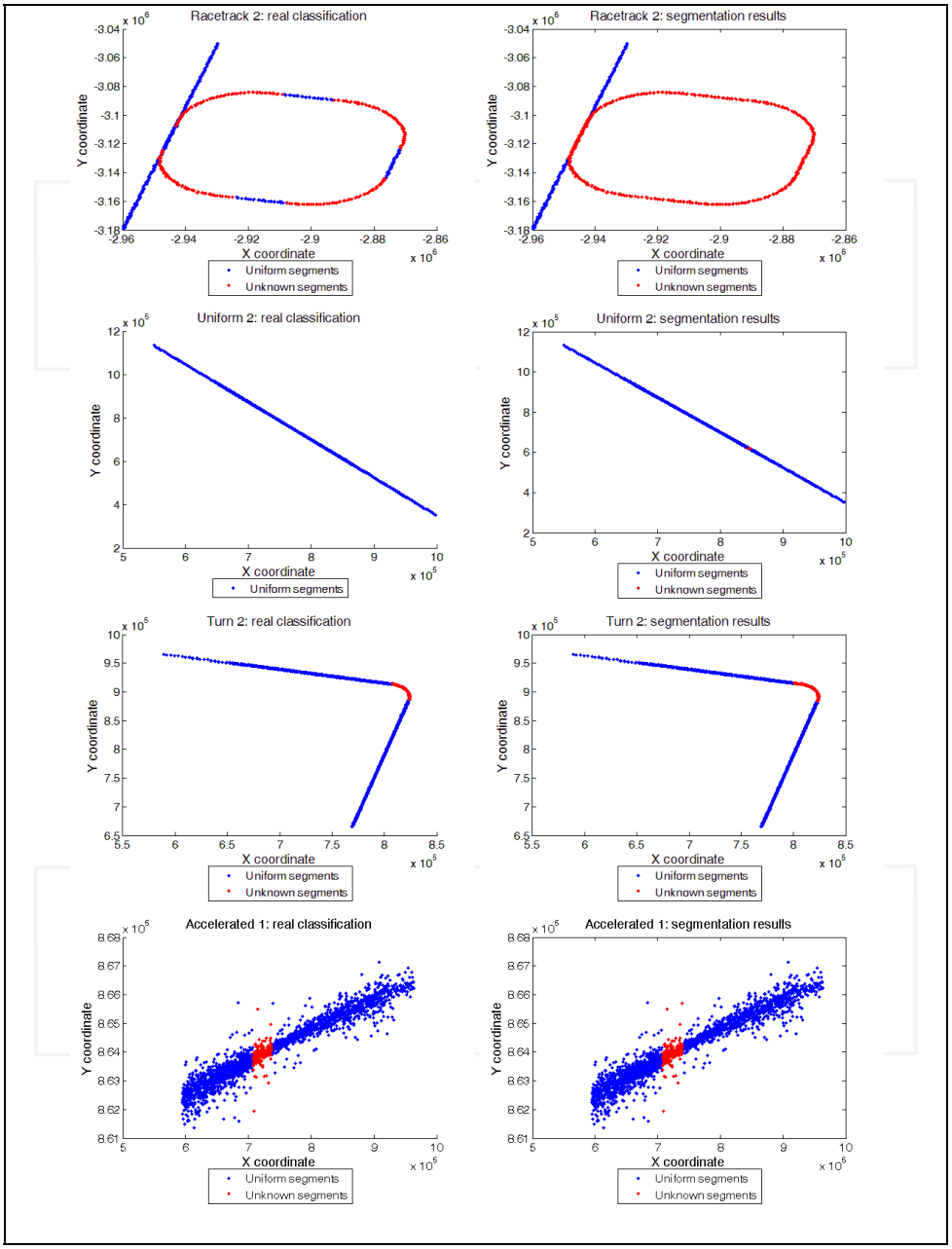

Fig. 9. Segmentation results overview

8. Conclusions

The automation of ATC systems is a complex issue which relies on the accuracy of its low level phases, determining the importance of their validation. That validation is faced in this work with an inherently offline processing, based on a domain transformation of the noisy measurements with three different motion models and the application of machine learning and filtering techniques, in order to obtain the final segmentation into these different models. This work has analyzed and defined in depth the uniform motion model and the algorithm's performance according to this model. The performance analysis is not trivial, since only one of the motion models in the algorithm is presented and the results obtained are, thus, only partial. Even so, these results are encouraging, having obtained good TPR and FPR values in most trajectory types, and a final number of segments which are reasonably similar to the real ones expected. Some issues have been pointed out, such as the behaviour of measurements belonging to uniform motion models when they are close to two different non-uniform segments (a typical situation during racetrack trajectories), but the final algorithm's results are required in order to deal with these issues properly. Future lines include the complete definition of the algorithm, including the non uniform motion models and the study of possible modifications in the domain transformation, in order to deal with the introduced difficulties, along with the validation with real trajectories.

9. References

Allchin, D.,(2001) "Error Types", *Perspectives on Science*, Vol.9, No.1, Pages 38-58. 2001.

Baxes, G.A. *(1994) "Digital Image Processing. Principles & Applications",* Wiley and Sons

Coello, C. A., Lamont, G. B., Van Veldhuizen, D. A. (2007) "*Evolutionary Algorithms for Solving Multi-Objective Problems*" 2nd edition. Springer

Dellaert, F. (2002) "*The Expectation Maximization Algorithm*". Technical Report number GIT-GVU-02-20. College of Computing, Georgia Institute of Technology

Drouilhet, P.R., Knittel, G.H., Vincent, A., Bedford, O. (1996) "*Automatic Dependent Surveillance Air Navigation System*". U.S. Patent n. 5570095. October, 1996

Eibe, F. (2005) "*Data Mining: Practical Machine Learning Tools and Techniques*". Second Edition. Morgan Kauffman

Famili, A., Sehn, W., Weber, R., Simoudis, E. (1997) "Data Preprocessing and Intelligent Data Analysis" *Intelligent Data Analysis Journal*. 1:1-28, March, 1997.

Fawcett, T. (2006) "An introduction to ROC analysis". *Pattern Recognition Letters*, 27. Pages: 861-874. International Association for Pattern Recognition

Garcia, J.; Perez, O.; Molina, J.M.; de Miguel, G.; (2006) "*Trajectory classification based on machine-learning techniques over tracking data*". Proceedings of the 9th International Conference on Information Fusion. Italy. 2006.

Garcia, J., Molina, J.M., de Miguel, G., Besada, A. (2007) "*Model-Based Trajectory Reconstruction using IMM Smoothing and Motion Pattern Identification*". Proceedings of the 10th International Conference on Information Fusion. Canada. July 2007.

Garcia, J., Besada, J.A., Soto, A. and de Miguel, G. (2009) "Opportunity trajectory reconstruction techniques for evaluation of ATC systems". *International Journal of Microwave and Wireless Technologies.* 1 : 231-238

Guerrero, J.L and Garcia J. (2008) "*Domain Transformation for Uniform Motion Identification in Air Traffic Trajectories*" Proceedings of the International Symposium on Distributed Computing and Artificial Intelligence (Advances in Soft Computing, Vol. 50), pp. 403-409, Spain, October 2008. Springer

Guerrero, J.L., Garcia, J., Molina, J.M. (2010) "*Air Traffic Control: A Local Approach to the Trajectory Segmentation Issue*". Proceedings for the Twenty Third International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems, part III, Lecture Notes in Artificial Intelligence, Vol. 6098, pp. 498-507. Springer

Gurney, K. (1997) "*An introduction to Neural Networks*". CRC Press.

Jensen, F.B., Graven-Nielsen, T. (2007) "*Bayesian Networks and Decision Graphs*". Second edition. Springer.

Kay, S.M. (1993) "*Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*". Prentice Hall PTR.

Kennedy D., Gardner, A. B. (1998) "Tools for analysing the performance of ATC surveillance radars", *IEE Colloquium on Specifying and Measuring Performance of Modern Radar Systems*. March, 1998. 6/1-6/4.

Keogh, E, Chu, S., Hart, D., Pazzani, M. (2003) "Segmenting Time Series: A Survey and Novel Approach". In: *Data Mining in Time Series Databases*, second edition.. pp 1-21. World Scientific

Mann, R. Jepson, A.D. El-Maraghi, T. (2002) "*Trajectory segmentation using dynamic programming*". Proceedings for the 16th International Conference on Pattern Recognition. 2002.

Meyer, P. (1970) "*Introductory Probability and Statistical Applications*" Second edition. Addison Wesley.

Pérez, O., García, J., Molina, J.M. (2006) "*Neuro-fuzzy Learning Applied to Improve the Trajectory Reconstruction Problem*". Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA06). Australia, November 2006. IEEE Computer Society.

Picard, R.; Cook, D. (1984). "Cross-Validation of Regression Models". *Journal of the American Statistical Association* 79 (387). Pages 575–583.

Quinlan, J.R. (1993) "*C4.5: Programs for Machine Learning*". Morgan Kaufmann

Rish, I. (2001) "*An empirical study of the naive Bayes classifier*". IJCAI 2001 : Workshop on Empirical Methods in Artificial Intelligence.

Shipley, R. (1971) "Secondary Surveillance Radar in ATC Systems: A description of the advantages and implications to the controller of the introduction of SSR facilities". *Aircraft Engineering and Aerospace Technology*. 43: 20-21. MCB UP Ltd.

Wickens, C.D., Mavor, A.S., Parasuraman, R. and McGee, J. P. (1998) "*The Future of Air Traffic Control: Human Operators and Automation*". The National Academies Press, Washington, D.C.

Yang, Y.E. Baldwin, J. Smith, A. Rannoch Corp., Alexandria, VA. (2002) "*Multilateration tracking and synchronization over wide areas*". Proceedings of the IEEE Radar Conference. August 2002. IEEE Computer Society.

Yin, L., Yang, R., Gabbouj, M., Neuvo, Y. (1996) "Weighted Median Filters: A Tutorial", *IEEE Trans. on Circuits and Systems*, 43(3), pages. 157-192.