Teaching of an Acrobatic Maneuver to an Aerial Robot

Semester Project

Adrian Arfire

Microengineering

Assistants:

Seyed Mohammad Khansari Zadeh

Learning Algorithms and Systems Laboratory

Prof. Aude Billard

EPFL

School of Engineering

Teaching of an Acrobatic Maneuver to an Aerial Robot

Adrian Arfire

Semester Project

June 2009

Abstract

In this project, it is desired to learn different acrobatic maneuvers through a set of demonstrations shown by an expert pilot.

Due to the complexity of the vehicle's dynamics, at first step, it is necessary to find an appropriate set of states that can best represent an acrobatic maneuver. Next, a change of frame of reference from the East-North-Up Coordinates System to Aircraft-Body Coordinates System is applied on the whole demonstration dataset to give a more accurate definition of the maneuver and to handle the problem associated with different starting positions and orientations. To increase the performance of the algorithm, the demonstration data points are filtered and refined.

After data preprocessing, the whole motion is encoded using Gaussian Mixture, and finally, an analysis of the model performance is made together with a discussion on the ways in which such a model could be used to control the aircraft.

Contents

Chapter 1 – Introduction

Programming by demonstration (PbD) is a technique for teaching a computer or a robot new behaviors by demonstrating the task to transfer directly, instead of programming it through machine commands.

In the Learning Algorithms and Systems Laboratory (LASA), this method has been widely used to teach different dynamical tasks (e.g. writing alphabets, putting an object into a container, wiping a tray, etc.) to humanoid robots. In this project, it is desired to extend the applicability of the current framework to aerial robots, and to evaluate its performance for general aerial maneuvers including “loop”, “level turn”, etc.

The flight platform

The aerial robot used for the project is a flying wing, developed by the Laboratory of Intelligent Systems (LIS) at EPFL, called a Micro-Air Vehicle (MAV). It was originally created as part of the SMAVNET program [1], which was aiming at creating an emergency wireless communication network through the use of swarming MAV’s.

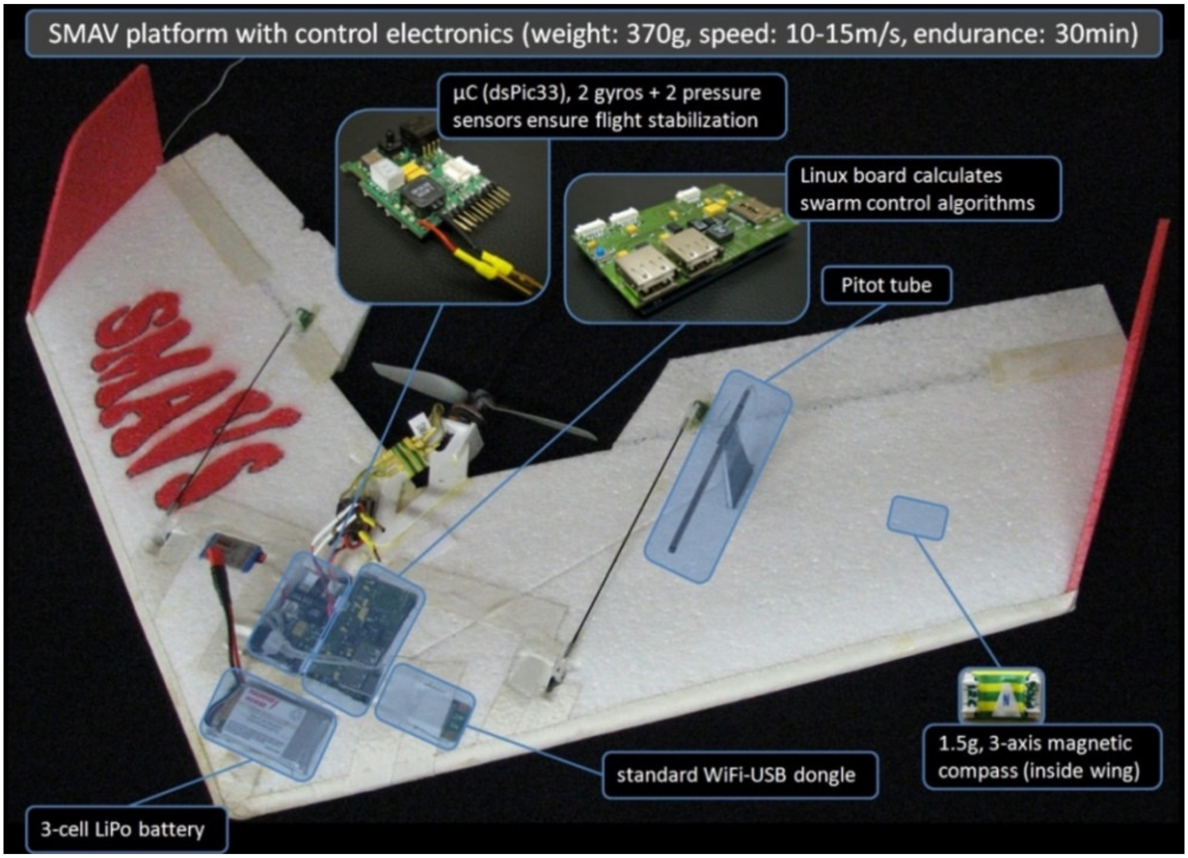
SMAV platform with control electronics (weight: 370g, speed: 10-15m/s, endurance: 30min)
µC (dsPic33), 2 gyros + 2 pressure sensors ensure flight stabilization
Linux board calculates swarm control algorithms
Pitot tube
SMAVS
1.5g, 3-axis magnetic compass (inside wing)
standard WiFi-USB dongle
3-cell LiPo battery

Figure 1: The MAV platform [1]

It is a custom-made flying wing type aircraft made of flexible EPP foam, electric motor (mounted behind the wing), 80cm wingspan, 350-400g, 30min flight time. The electric components like battery, motor, motor-controller and servos are standard parts from remote controlled hobby airplanes. The platform hardware's simplicity and low cost are key elements to enable flying swarm systems.

The core sensors are no more than four: a differential (Freescale MPXV5004DP) and an absolute pressure sensor (Freescale MPXH6115A) permit to determine airspeed and altitude, 2 rate gyros (Analog Devices ADXRS610) measure the plane's rotational speed around the yaw and pitch axes. The basic flight control strategy uses raw sensor values and is able to robustly steer the plane to autonomously take off, reach a predetermined altitude, circle with a constant turning rate, and land after a given time or when triggered [2].

The *Aeropic* flight controller uses a Microchip microcontroller (dsPIC33F) which gives the commands to the throttle motor and the left/right servos which control the elevons (the combination of ailerons and elevators) of the MAV. The system communicates with the ground station (Ishtar) via an XBee Pro radio modem module.

In addition to the basic sensors presented above, for this project, the MAV was fitted with a GPS module and an Inertial Measurement Unit (IMU).

Project Objective

The goal of this project is to implement a method of teaching a general maneuver to the MAV platform. In order to achieve this, demonstrative flights need to be performed by a ground pilot from which a model for each type of maneuver needs to be extrapolated.

In the following chapters, a set of procedures to preprocess the demonstration data are first discussed. The results of using Gaussian Mixture for modeling the maneuvers are then presented together with an analysis of the obtained model.

The final part of this paper represents a discussion on the utility of such a model for flight control.

Chapter 2 – Processing the Demonstration Data

In order to have an accurate procedure of obtaining the underlying maneuver, it is necessary to perform some preprocessing on the data set before encoding the model, and then to define an appropriate set of aircraft's states.

At first, the basic signal processing that is done on the rough data is presented. In the second subsection, the way the demonstration data should be represented is discussed. Also here, a method of automatically extracting the maneuver data from a set of flight data is suggested. Finally, the method of aligning the data is presented.

Interpolation and smoothing

The first issue that needs to be handled is the interpolation of the demonstration data in order to handle the fact that some data packets are always lost during recording. The interpolation step is chosen to be equal to the timing of the main loop of the flight controller: 65ms.

A smoothing filter is then used to clean some of the sensor noise. This is a standard 5-point moving average filter [3].

Apart from this basic signal processing another function is implemented to remove the discontinuities from the IMU readings, which are caused by the fact that the Euler angles are, by standard, defined over a bounded interval. This is done by detecting the discontinuities and offsetting the subsequent data appropriately.

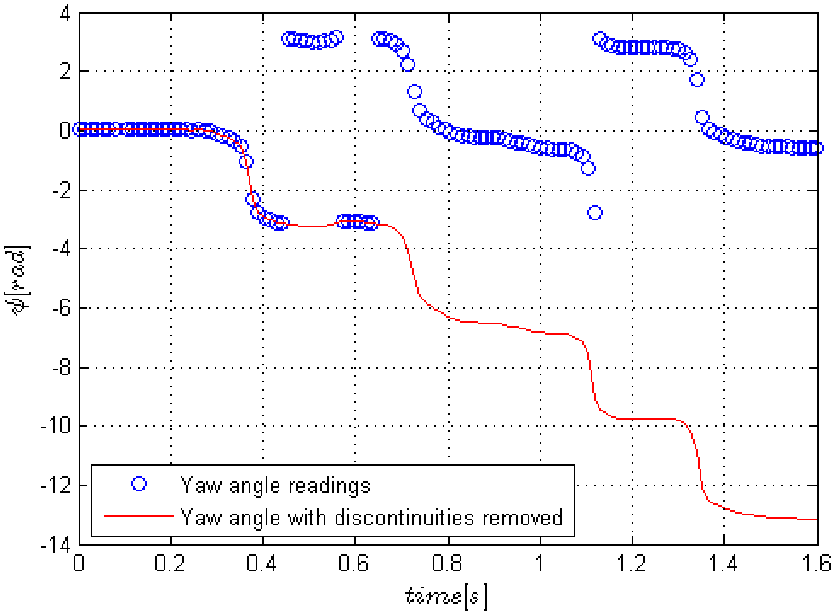

Figure 2: Discontinuity removal for Euler angle readings. The Yaw angle is defined over (-π, π)

Data representation and alignment

The issue of choosing a good way of looking at the underlying information about the maneuver, embedded in the flight data, and then properly aligning the different demonstrations is crucial in order to obtain a clear and coherent model of different maneuvers.

Changing the frame of reference

Due to the difficulty of piloting the aircraft from the ground it is clear that the demonstrations would not be properly geometrically aligned. For example, if one is to take a loop maneuver, there would always be a difference in the initial yaw of the maneuver, but also, seeing that the demonstrations are far from ideal loops, there would always be a deviation from the ideal vertical geometrical plane of a loop. In reality, the trajectories will not really be restricted to just one plane, but rather the loop will always have projections on the x-y and y-z planes also.

Continuing with the example of the loop maneuver, one might try to focus on the particularities of such a maneuver in order to align it. A loop's trajectory will always have two clear directions of relatively large variation i.e. two very significant principal components (that would be parallel to the x and z-axis for an ideal loop, with zero initial yaw).

The principal components would define a frame of reference tied to each loop demonstration and aligning these frames of reference might seem a good idea for aligning the maneuvers.

On a closer look it becomes apparent why this in fact would not solve our representation problem. Firstly, although a rotation of the trajectories around the z-axis (a yaw alignment) is easy to do, a rotation around the first principal component (a roll alignment) is impossible if we think about the fact that we don't desire to only align trajectories but also the commands that were given in order to obtain those trajectories. Without a prior model of the dynamics of the plane the implication on the commands of "rolling" the loop trajectories, in order to align them, remains unknown.

Secondly, it is clearly not a good idea to try to do an alignment based on the particular properties of a loop, considering that the goal is to model general maneuvers.

And finally, if we think about flight control during a maneuver, the position data is not as relevant to the control strategy as are the velocity values. This idea will be looked into with more details in the next section that presents a better way of representing the flight data.

To this point, the way the flight data has been looked at has been from the GPS's (x, y, z) frame, which is the East-North-Up (ENU) Coordinates System.

The local ENU Coordinates, used by the GPS, are formed from a plane tangent to the Earth's surface fixed to a specific location and hence it is sometimes known as a "Local Tangent" or "local geodetic" plane [4]. The axes are defined as:

- X Axis - Positive in the direction of East
- Y Axis - Positive in the direction of North (perpendicular to X Axis)
- Z Axis - Positive upwards (perpendicular to X-Y Plane)

In order to have a flight controller one needs to be able to control the six degrees of freedom of the aircraft: three of which describe its position and the other three describe its attitude. In stating this, it becomes apparent that looking at the aircraft’s flight from the ENU Coordinate System would lead to a rather awkward way of controlling it.

This becomes clear if we take an example: After demonstrating a series of loops along a certain direction to the aircraft, we wouldn’t want the system to only learn how to do a loop along that single direction, but rather along any arbitrarily chosen direction. The manner to achieve this is not obvious if we are teaching the maneuver in the ENU Coordinate System.

A better approach, which solves this issue, is to look at the flight from the aircraft’s perspective, i.e. to represent the data in the Aircraft-Body Coordinate System.

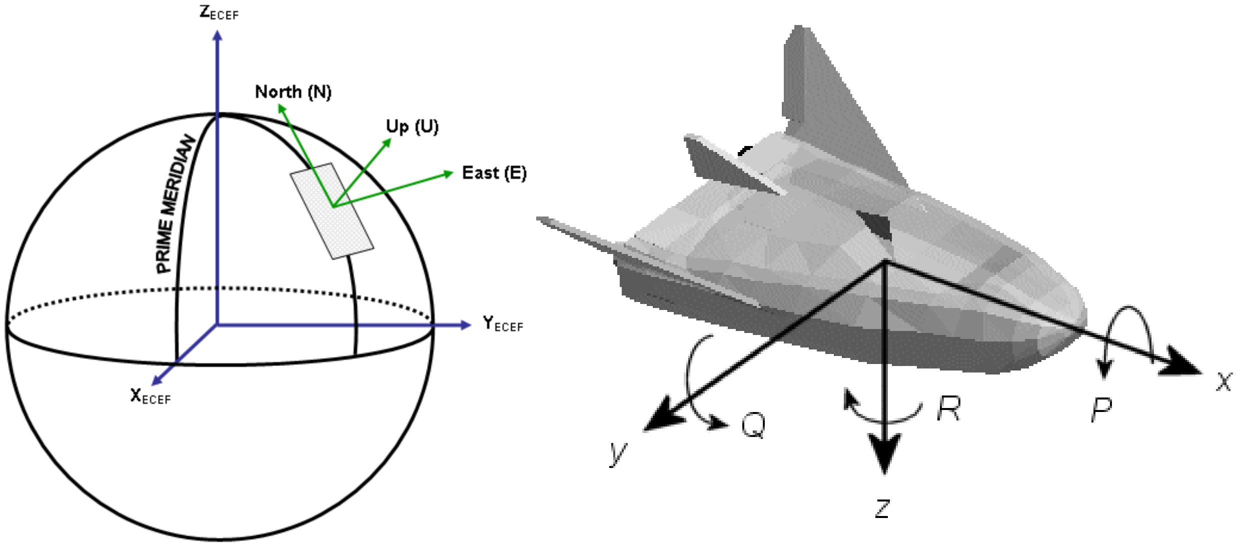

Figure 3: The ENU Coordinate System (left) [4] and the Aircraft-Body Coordinates System (right) [5]

The Aircraft-Body Coordinate System [6]

This coordinate system is defined by the Body Axes, which are based about the aircraft's center of gravity:

- X Axis - positive forward, through nose of aircraft
- Y Axis - positive to Right of X Axis, perpendicular to X Axis
- Z Axis - positive downwards, perpendicular to X-Y plane

As previously stated, for the control of the aircraft we need to look at controlling its velocity. In the Aircraft-Body Coordinate System the speed of the aircraft is projected into the following components:

- u – the speed along the X axis
- v – the speed along the Y axis
- w – the speed along the Z axis

By controlling the velocity over time we can control the aircraft position. This is of course not enough as we also need to control the attitude of the aircraft during the maneuver.

The attitude of the aircraft is described by the attitude angles (also called Euler angles). They represent the angles between the inertial frame and the Aircraft-Body Coordinate System. If the ENU Coordinate System is considered as the inertial frame, then the definition of the Euler angles is the following:

- Roll (φ) - angle of Y Body Axis (wing) relative to the local tangential plane
- Pitch (ϑ) - angle of X Body Axis (nose) relative to the local tangential plane
- Yaw (ψ) - angle of X Body Axis (nose) relative to East

One can either use the Euler angles directly in order to describe the attitude of the airplane or the rate of change of these angles can be used (as in the case of the position). They are:

- $\dot{\varphi}$ – roll rate
- $\dot{\vartheta}$ – pitch rate
- $\dot{\psi}$ – yaw rate

The Euler angle rates can be related to the angular velocities (p, q, r) in the Aircraft-Body Coordinates through the following transformation:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin(\vartheta) \\ 0 & \cos(\varphi) & \cos(\vartheta)\sin(\varphi) \\ 0 & -\sin(\varphi) & \cos(\vartheta)\cos(\varphi) \end{bmatrix} \begin{bmatrix} \dot{\varphi} \\ \dot{\vartheta} \\ \dot{\psi} \end{bmatrix} \qquad (2.1)$$

The values of (*p*, *q*, *r*) can be obtained directly from the rate gyros onboard the MAV. It should be noted here that there are two sets of rate gyros on the MAV. The first one is the standard gyros on the Aeropic flight controller and the second one is the set included in the IMU module. For this project the rate gyros in the IMU are considered.

Sometimes in flight controllers both Euler angles and angular velocities are used, because, if they are obtained from different measurement systems, this can give a greater reliability to the flight controller. The issue of whether both Euler angles and angular velocities should be used in the model, or only one of these two sets of parameters, will be discussed in Chapter 3.

Transforming the recorded data

The available sensors on board the MAV don't give us direct values of the (*u*, *v*, *w*) components. Nevertheless, these values can be obtained by applying a transformation to the velocity components (v_x, v_y, v_z) from the ENU Coordinate System as follows:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = T(\varphi, \vartheta, \psi) \cdot \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \qquad (2.2)$$

$$v_x = \dot{x}, v_y = -\dot{y}\,, v_z = -\dot{z} \qquad (2.3)$$

$$T(\varphi, \vartheta, \psi) = \begin{bmatrix} \cos(\vartheta)\cos(\psi) & -\cos(\vartheta)\sin(\psi) & \sin(\vartheta) \\ \cos(\varphi)\sin(\psi) + \sin(\varphi)\sin(\vartheta)\cos(\psi) & \cos(\varphi)\cos(\psi) - \sin(\varphi)\sin(\vartheta)\sin(\psi) & -\sin(\varphi)\cos(\vartheta) \\ \sin(\varphi)\sin(\psi) - \cos(\varphi)\sin(\vartheta)\cos(\psi) & \sin(\varphi)\cos(\psi) + \cos(\varphi)\sin(\vartheta)\sin(\psi) & \cos(\varphi)\cos(\vartheta) \end{bmatrix} \text{(2.4)[7]}$$

The Euler angle values are given directly by the on board IMU. The velocity components in the ENU Coordinates System are available by derivation by time of the GPS readings.

The minus sign in front of the z and y components is due to the fact that in ENU coordinates the z axis is upwards and y axis to the left, while in Aircraft-Body Coordinates the z axis is downwards and the y axis to the right.

Maneuver extraction

In this section, a possible method for extracting maneuvers from longer sets of recorded data, in which one or more maneuvers might be present, is put forth. This will be illustrated on a set of data containing two loops.

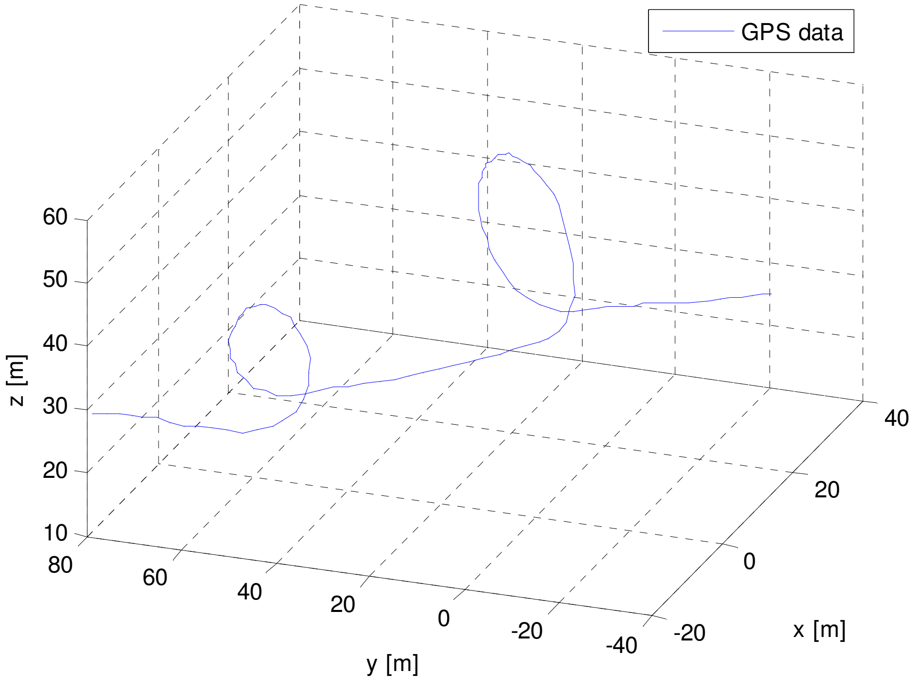

Figure 4: Example of demonstration containing 2 loops

The technique consists of two parts: a dynamic threshold operation and a morphological opening operator.

Dynamic threshold

This operation is applied on the first derivatives of the Euler angles, selecting only those data points for which the derivative of the Euler angles are higher than the respective threshold, for at least one of the three time series: $\dot{\varphi}(t)$, $\dot{\vartheta}(t)$ or $\dot{\psi}(t)$. In other words, after selecting for each of the three time series in relation to their respective threshold, the three resulting selection signals are subjected to a logical OR operation so as to obtain a single selection signal.

For implementing the threshold operation, a 2-bin Lloyd-Max quantization [3] is used for each of the above signals. This results in a rough selection of the data of interest, as can be seen in the following graphs.

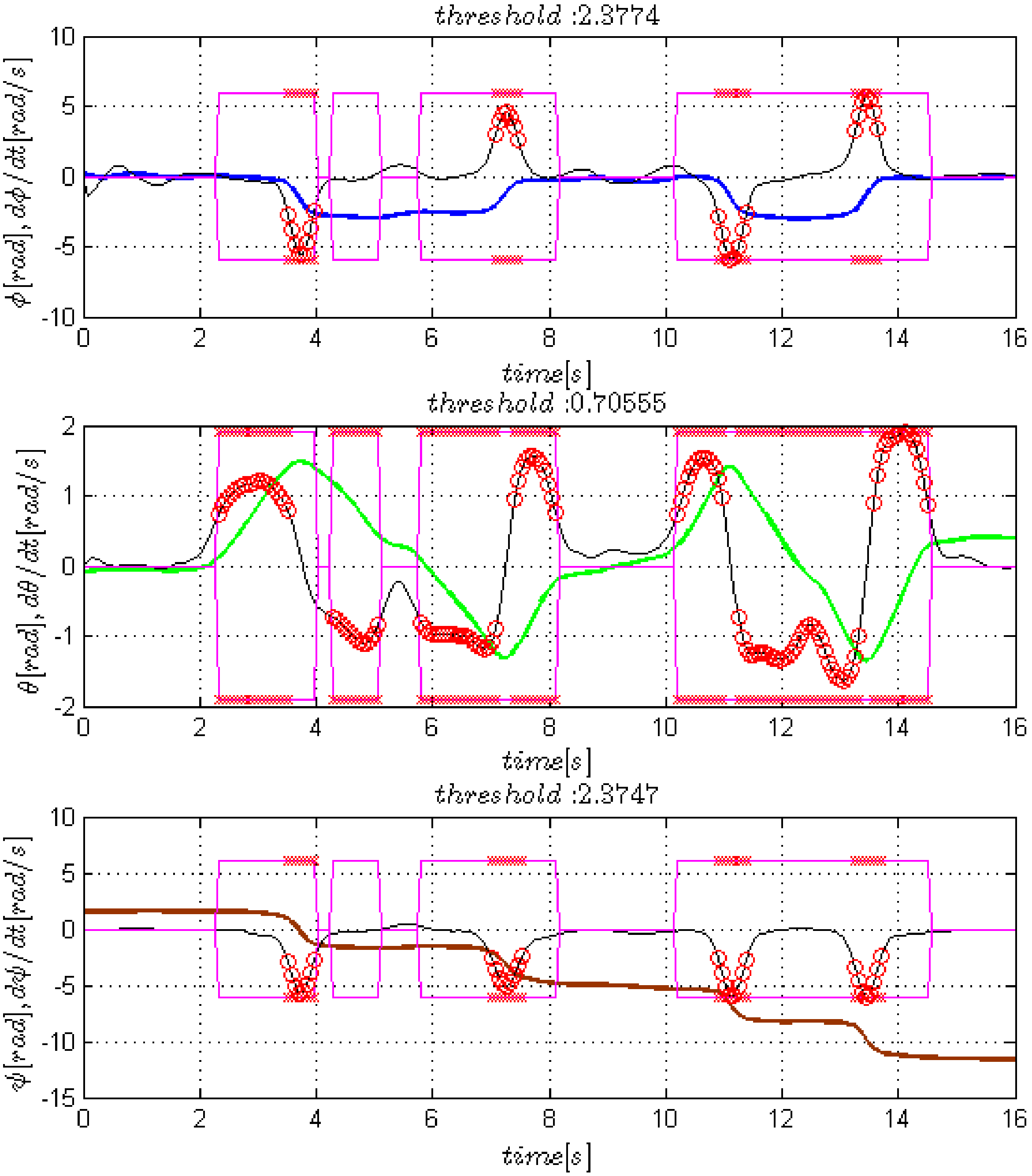

Figure 5: Selection after threshold. The selection for each of time series is shown with red markers and the selection that results after the OR operation is shown in magenta

Morphological opening

In order to refine the selection which is, in fact, a binary signal, a morphological opening [3] is used. In mathematical morphology, opening is the dilation of the erosion of a set A by a structuring element B:

$$A \circ B = (A \ominus B) \oplus B \qquad (2.5)$$

where $\ominus$and $\oplus$ denote erosion and dilation, respectively.

Erosion

The erosion of the binary signal *A* by the structuring element *B* is defined by:

$$A \ominus B = \{x \in \mathbb{E} | (B)_x \subseteq A\} \qquad (2.6)$$

where $\mathbb{E}$ represents the support of the signal and the structuring element $(B)_x$ represents, in this case, a segment with the origin, which is set in the center of the segment, being *x*.

The above definition can be read as follows: For the given A and B, by applying erosion we keep only those points x of the support $\mathbb{E}$ of A, for which the structuring element centered in x is contained in its entirety in A.

The effect of this operator is, as the name suggests, an erosion of the selection signal. Its purpose is to clean possible false selections, due to short variations of the Euler angles that are inconsistent with a maneuver, but might still be higher than the threshold.

In the case of the data used for this example there are no such false selections so the only effect is seen in the fact that the holes in the selections get greater. Even though in this case the erosion does not give a clear improvement it is easy to see how for other sets of data, with short, but significant perturbations in the flight trajectory, it would.

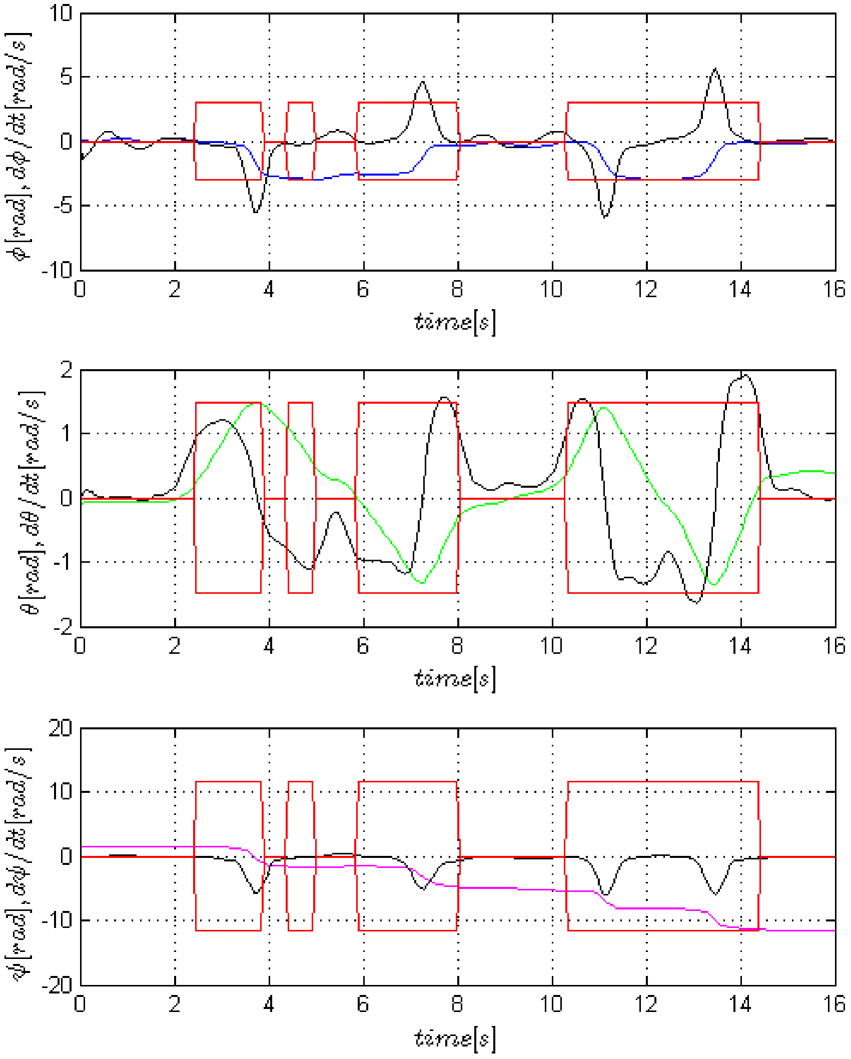

Figure 6: Selection after erosion

Dilation

The dilation of *A* by the structuring element *B* is defined by:

$$A \oplus B = \cup_{x \in A}(B)_x \qquad (2.7)$$

where the structuring element is defined like in the case of the erosion.

This operation has the effect of filling all the holes in the selection "signal" and giving a clean selection of the two loops in this example.

The advantage of using this method is an automatic extraction of the data of interest which is helpful also in the process of aligning the demonstrations, since only the relevant data is kept.

On the other hand, the big weakness of the implementation of the morphological opening at this time is the fact that it is not dynamical. Fixed sizes are used for the structuring elements of the erosion and the dilation, which were chosen by several trials on the available data, but no guaranty can be given that it will work on all demonstrations.

Nevertheless, if this issue will be resolved, the technique would help to speed up the processing of the demonstration data.

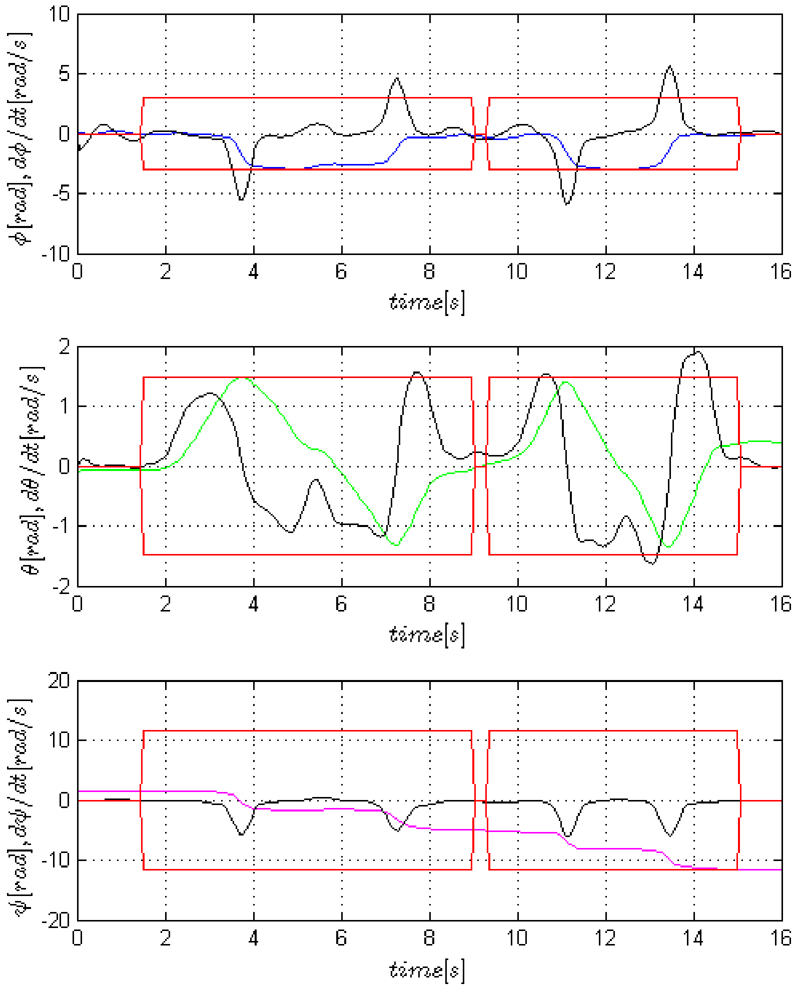

Figure 7: Final selection, after dilation

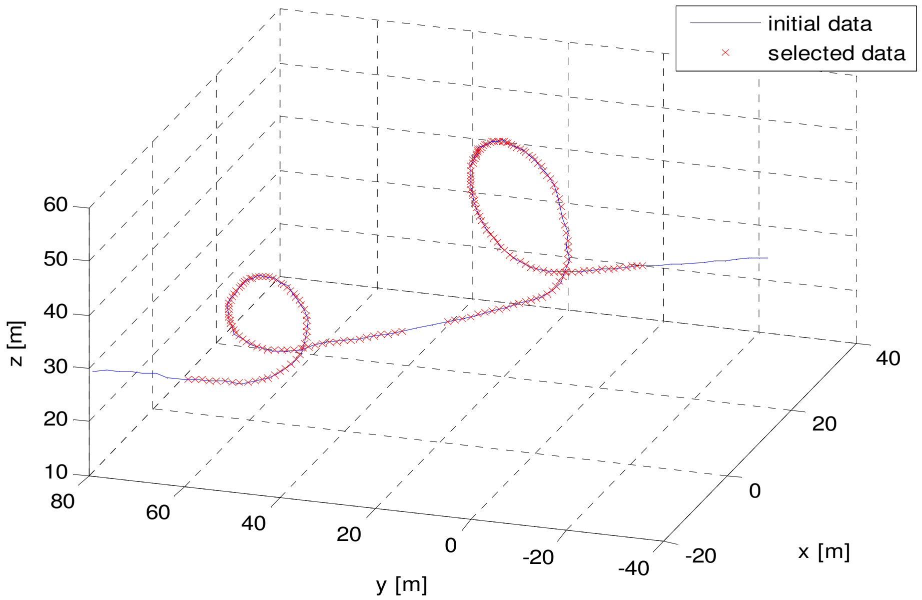

Figure 8: View of selection in the (x, y, z) coordinates

Demonstrations alignment

In order to align all the demonstration sets for a particular maneuver, an alignment along the time axis is chosen.

For this purpose, a cost function *J* is defined. Being given the time series of sensor readings ($x_S(t)$) and the time series of commands ($u_S(t)$), the cost function *J* is computed as:

$$J(t) = x_S^T(t)Qx_S(t) + u_S^T(t)Ru_S(t) \qquad (2.8)$$

The matrices *Q* and *R* need to be positive definite, and for simplicity they are chosen diagonal. The values on the diagonal are chosen as such as to define the priority for each dimension. Furthermore, it can also handle the problem of having different scales for each state. This is important since, for example, the commands, which have values in the range of the hundreds and of the thousands, would leave the other data with no real influence over the cost function, seeing that their values usually don't even reach 30.

The components of $u_S(t)$ are:

$$u_S(t) = \begin{bmatrix} \delta T(t) \\ \delta A(t) \\ \delta E(t) \end{bmatrix} \qquad (2.9)$$

where $\delta T(t)$ is the throttle command, $\delta A(t)$ is the ailerons command and $\delta E(t)$ represents the elevators command at time t.

In the case of the sensor data the composition of $x_S(t)$ depends on whether we choose to use both the Euler angles and the angular velocities or just one of the two sets.

In the following example, where five demonstrations of a loop maneuver are considered, only the results obtained by running the alignment algorithm with no influence from the angular velocities are presented as the results obtained by disregarding them are slightly better. The rest of the components of Q and R are chosen in such a manner as to scale the data by the maximum absolute value that is observed. This, of course, means that all dimensions are given equal importance.

$$Q = diag\left(\frac{1}{\max(|u|)}, \frac{1}{\max(|v|)}, \frac{1}{\max(|w|)}, 0,0,0, \frac{1}{\max(|\varphi|)}, \frac{1}{\max(|\vartheta|)}, \frac{1}{\max(|\psi|)}\right)^2 \tag{2.10}$$

$$R = diag\left(\frac{1}{\max(|\delta T|)}, \frac{1}{\max(|\delta A|)}, \frac{1}{\max(|\delta E|)}\right)^2 \tag{2.11}$$

Although we have opted for simplicity to give the same importance to all the dimensions of our data sets, a differentiated weighing of the influence of a certain command or a certain sensor reading could prove advantageous in some cases.

After the cost function is computed, one of demonstration sets is arbitrarily chosen and all the others are aligned with respect to it. This is done in the following way: the rest of the demonstration sets are translated along the time axis and the difference between the value of the cost function of the fixed demonstration and of the one that is being shifted is computed for each time sample. All of these differences are then summed up. By minimizing this sum, the distance between the cost functions is minimized and the demonstrations are aligned along time.

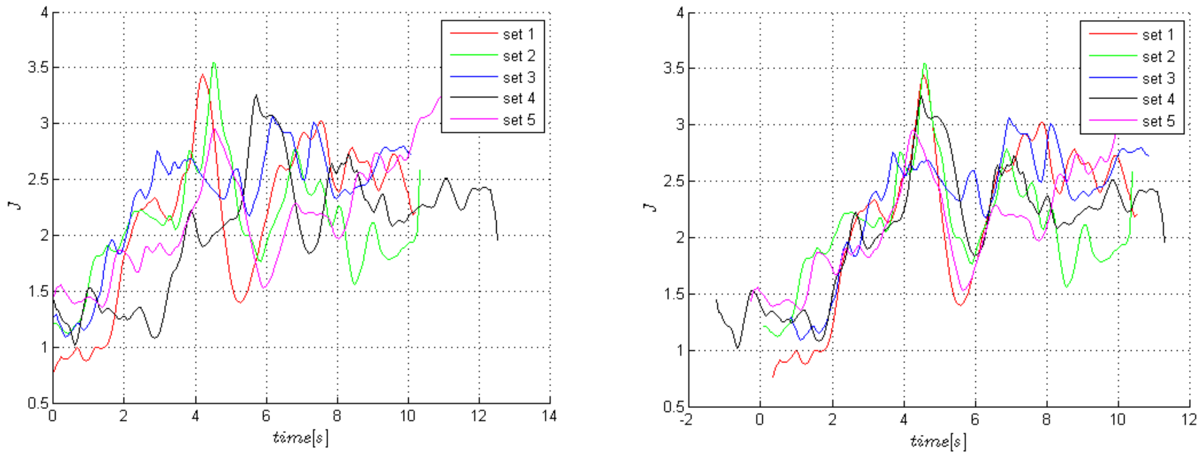

Figure 9: Cost functions before (left) alignment and after (right) alignment

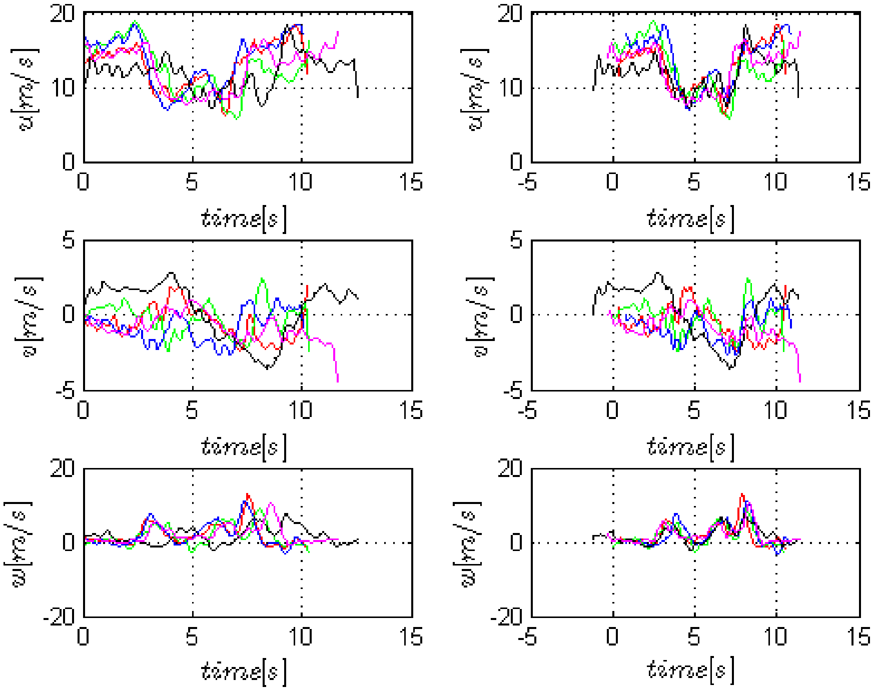

Figure 10: u, v, w components before (left) and after (right) alignment

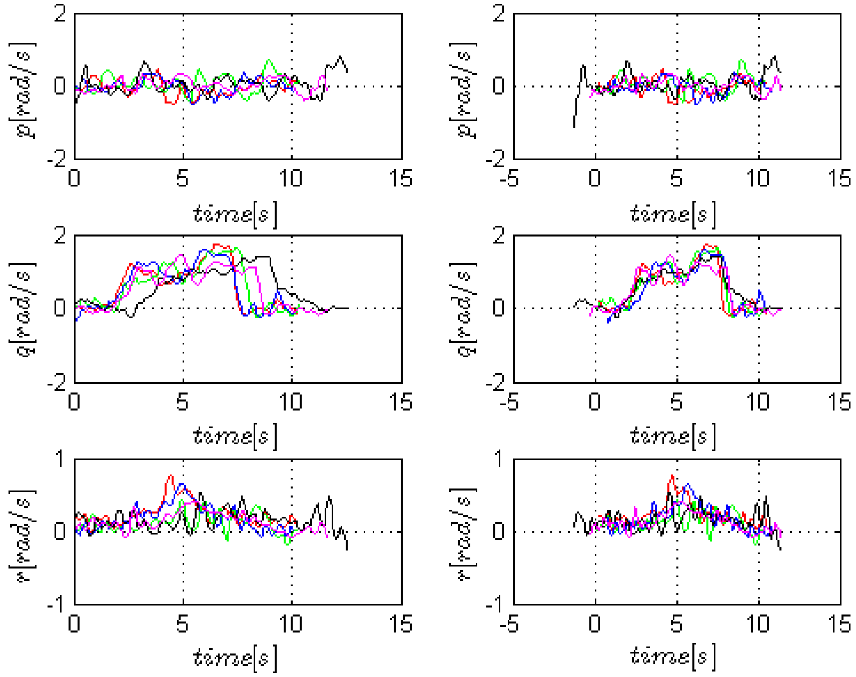

Figure 11: p, q, r components before (left) and after (right) alignment

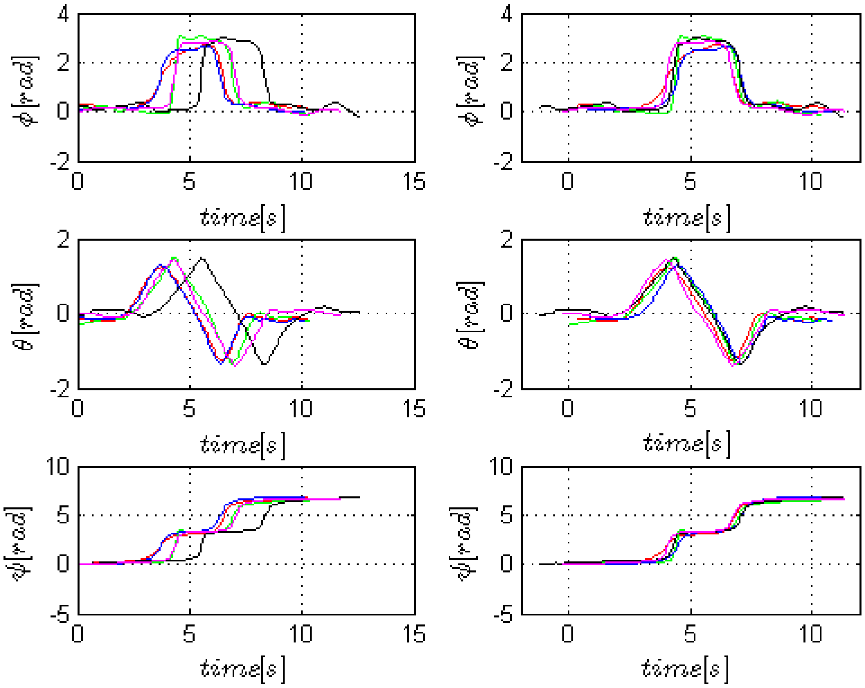

Figure 12: ϕ, θ, ψ components before (left) and after (right) alignment

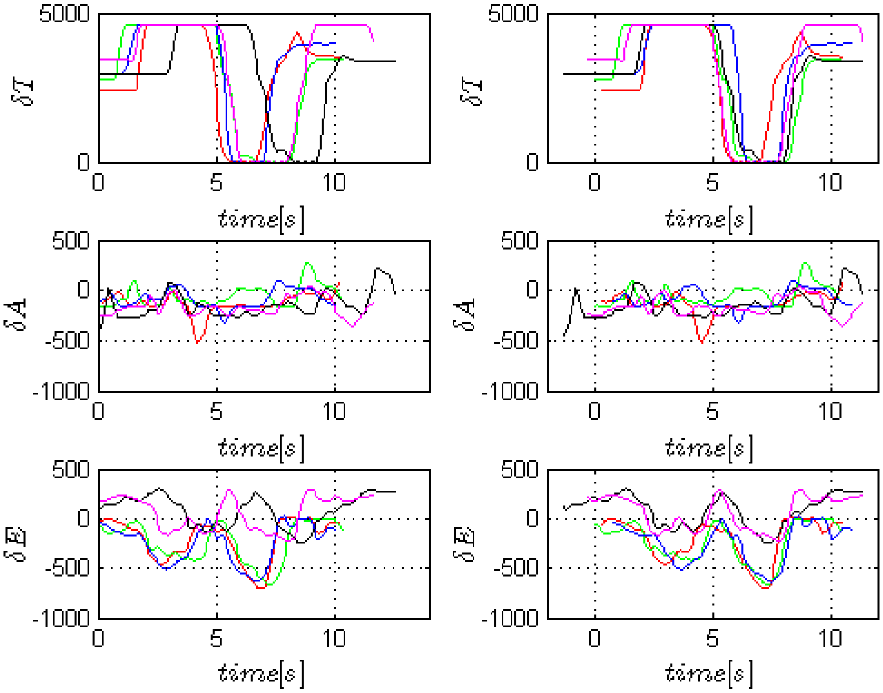

Figure 13: Commands before (left) and after (right) alignment

Chapter 3 - The Gaussian Mixture Model

After aligning the data the next step is to use Expectation Maximization (EM) to model the process by a mixture of Gaussian components. For this project a version of the EM algorithm implemented in LASA [8] is used.

In this chapter several issues related to the Gaussian Mixture model are discussed including: scaling the command values, choosing the number of Gaussian components of the model, an analysis of the model's performance and selecting the dimensionality of the model.

Scaling the commands

Because of the very large difference of scale between the range of values taken by the commands in comparison with the other parameters, scaling them down was a crucial step before using the EM algorithm.

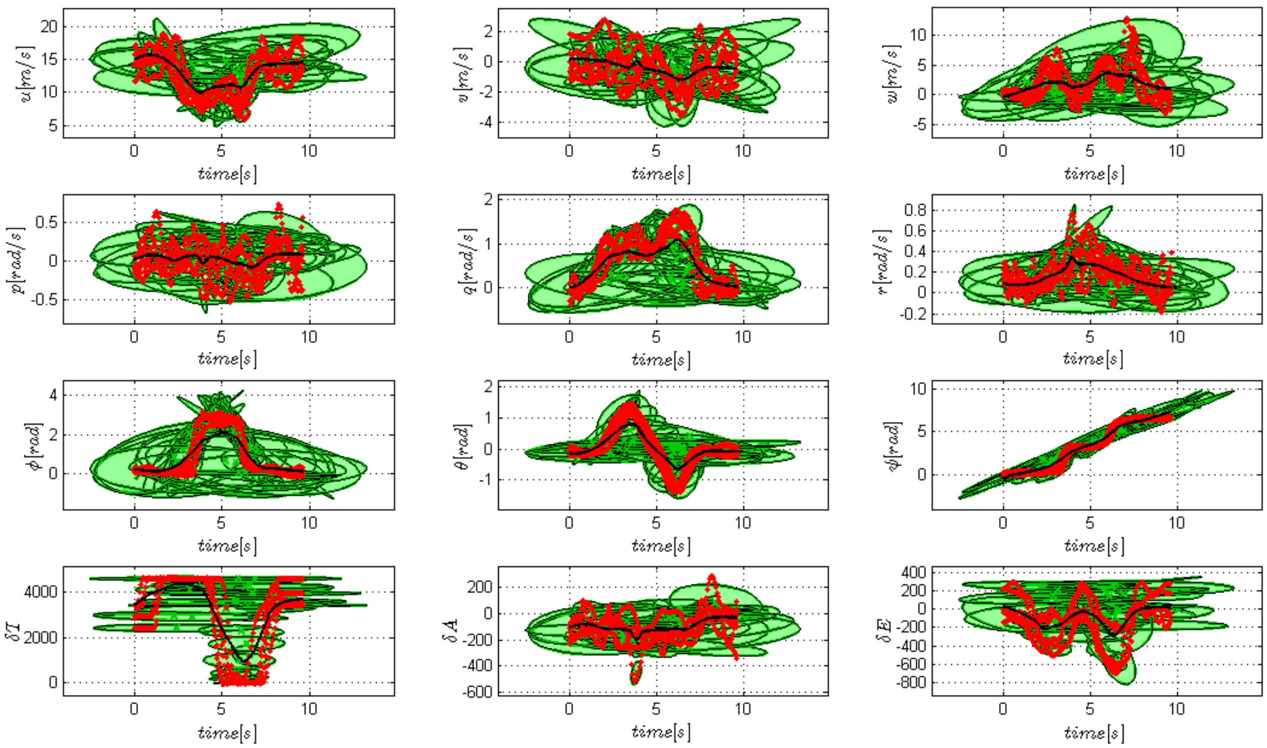

Figure 14: GMM without scaling down the commands

When this was not done the resulting Gaussian fitting is unacceptable, as can be seen in the figure 13, where the regression signal for many of the parameters is very far from the mean of the demonstration data. The most affected estimations are those of the Euler angles.

After scaling the throttle values by 1000 and the ailerons and elevators commands by 100, the results are clearly superior.

The number of Gaussian components

In order to decide on the number of Gaussians in the model, the Bayesian Information Criterion (BIC) was used. This is a criterion for model selection among a class of parametric models with different numbers of parameters.

When estimating model parameters using maximum likelihood estimation, it is possible to increase the likelihood by adding additional parameters, which may result in overfitting. The BIC resolves this problem by introducing a penalty term for the number of parameters in the model.

For Gaussian Mixture Models (GMM) the BIC has the following formula [8]:

$$BIC = -\mathcal{L} + \frac{n_p}{2}\log(N) \tag{3.1}$$

where:

- $\mathcal{L}$ is the log-likelihood of the model
- n_p is the number of free parameters required for a mixture of K components:
$$n_p = (K-1) + K\left(D + \frac{1}{2}D(D+1)\right) \tag{3.2}$$
- N is the number of D-dimensional data points

An investigation was conducted to see the value of the BIC, for models learned from the same loop demonstrations as before, with up to 75 components. Another example, using level turn demonstrations, is included in the Annex.

Three cases were considered:

1) $D = 13$, with:

$$P = \begin{bmatrix} x_S \\ u_S \end{bmatrix}, where: x_S = [t\ \ u\ v\ w\ p\ q\ r\ \varphi\ \vartheta\ \psi]^T, u_S = \begin{bmatrix} \delta T \\ \delta A \\ \delta E \end{bmatrix} \tag{3.3}$$

2) $D = 10$, with:

$$P = \begin{bmatrix} x_S \\ u_S \end{bmatrix}, where: x_S = [t\ \ u\ v\ w\ p\ q\ r]^T, u_S = \begin{bmatrix} \delta T \\ \delta A \\ \delta E \end{bmatrix} \tag{3.4}$$

3) $D = 10$, with:

$$P = \begin{bmatrix} x_S \\ u_S \end{bmatrix}, where: x_S = [t\ \ u\ v\ w\ \varphi\ \vartheta\ \psi]^T, u_S = \begin{bmatrix} \delta T \\ \delta A \\ \delta E \end{bmatrix} \tag{3.5}$$

The result of this investigation can be seen in figure 14. In the first case, when all the possible dimensions are used, the BIC reaches a minimum for about 35 components. When the Euler angle are not used, the minimum is reached for about 50 components and in the last case, when the angular velocities are not used, the minimum is obtained for around 40 Gaussian components.

One of the first things that can be noticed is that these are very large values and probably a control algorithm using a GMM of 35 to 50 components would be unrealistic for the computational power of the dsPIC33F controller in the MAV. It should also be noted that the dsPIC33F does not have a floating point unit and, thus, all floating point operations are in fact software simulated. This is an issue that should be kept in mind if a controller based on a GMM with many components were to be implemented.

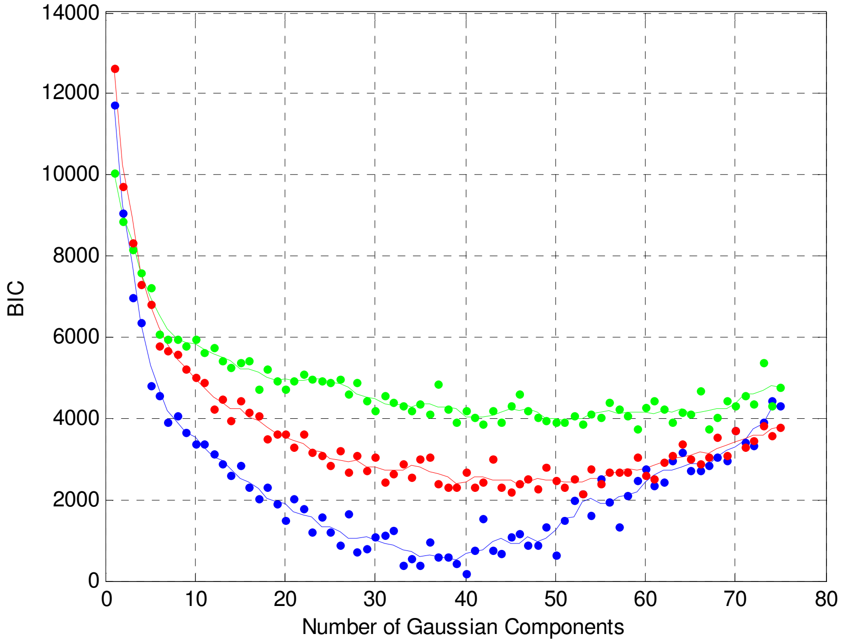

Figure 15: BIC for case 1 (blue), case 2 (green) and case 3 (red)

Another thing that can be observed is the fact that the BIC for the 13-dimensional case is smaller than for both of the other two cases, but, of course, a model with 13 dimensions would be computationally more expensive and has some further drawbacks that will be discussed in the next subchapter. Between the two 10-dimensional cases, the case in which the Euler angles are used scores better for both the number of components for which we obtain the minimum of the BIC and the value of the minimum in itself, which is lower than in the case in which the angular velocities are used.

Next, an analysis of the performance of the models obtained for the above cases is presented for the number of components that give the minimum of the BIC and also for a more realistic number of components: 15.

Performance of the GMM

In order to evaluate the performance of the GMM for each of the previously stated cases, the Root Mean Squared Error (RMSE) was computed for the regression signal obtained with the computed models using time as the only input and all the other states as outputs.

The RMSE is a frequently-used measure of the differences between values predicted by a model or an estimator and the values actually observed from the process being modeled or estimated. The definition of the RMSE is:

$$RMSE(\hat{\xi}) = \sqrt{MSE(\hat{\xi})} = \sqrt{E\left(\left(\hat{\xi} - \bar{\xi}\right)^2\right)} \qquad (3.6)$$

where $\bar{\xi}$ represents the mean of the specific parameter being evaluated computed for the demonstration sets. In the following graphs RMSE is referred by the E value.

Apart from the RMSE, the ratio between RMSE and the mean of the standard deviation ($\bar{\sigma}$) of the demonstration sets is computed for each dimension:

$$R = \frac{RMSE}{\bar{\sigma}} \qquad (3.7)$$

A value of R>3 would represent an unacceptable performance. This value is chosen in the idea that a performance that would keep the variance of the estimation into the 3σ band is required of the model.

Considering the first case (*D* = 13), with 35 Gaussian components the performance of the model is very good, the regression signal RMSE not crossing even the one standard deviation level.

After using the reverse of the transformation described in the previous chapter to get back to the Earth-Fixed Coordinate System and integrating over time the velocities, one can see that estimated trajectory has also a variance that is less than one standard deviation.

The following graphs illustrate this performance for both the chosen parameters of the model and the trajectory.

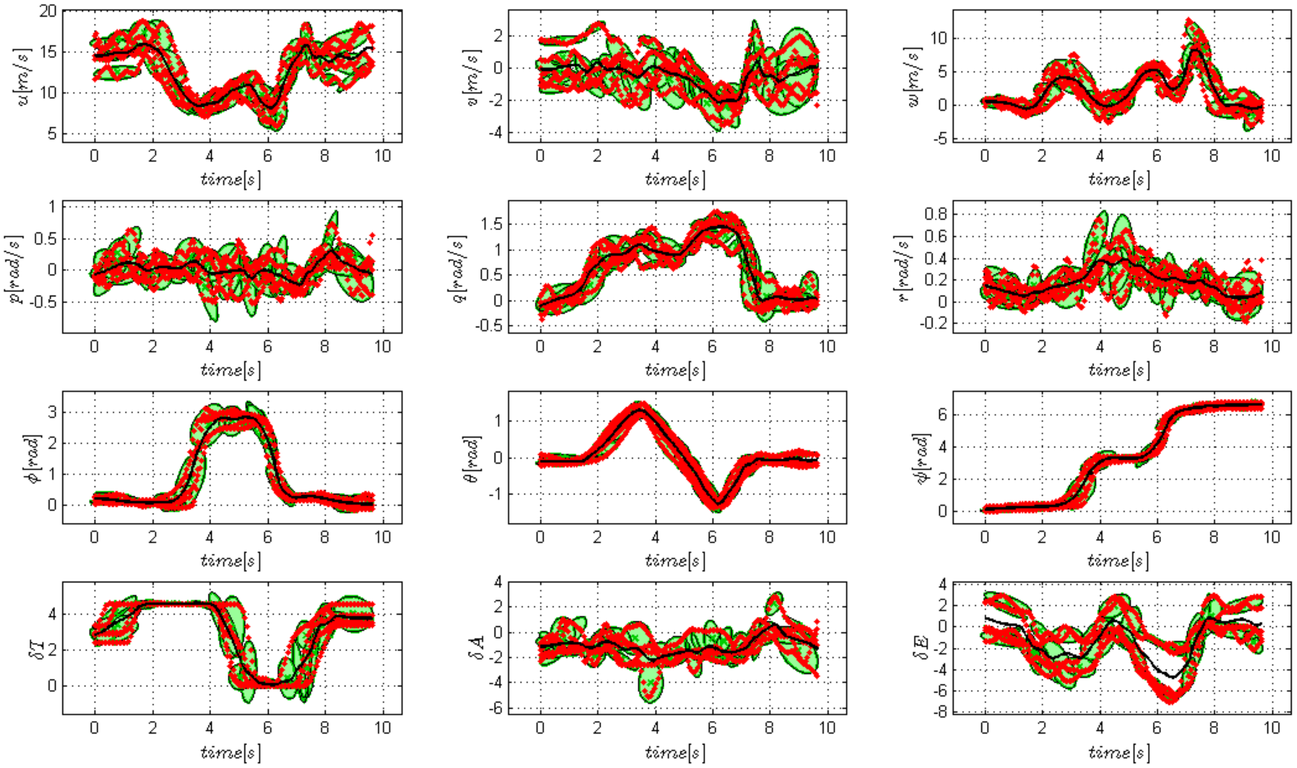

Figure 16: GMM and regression for 35 Gaussian components

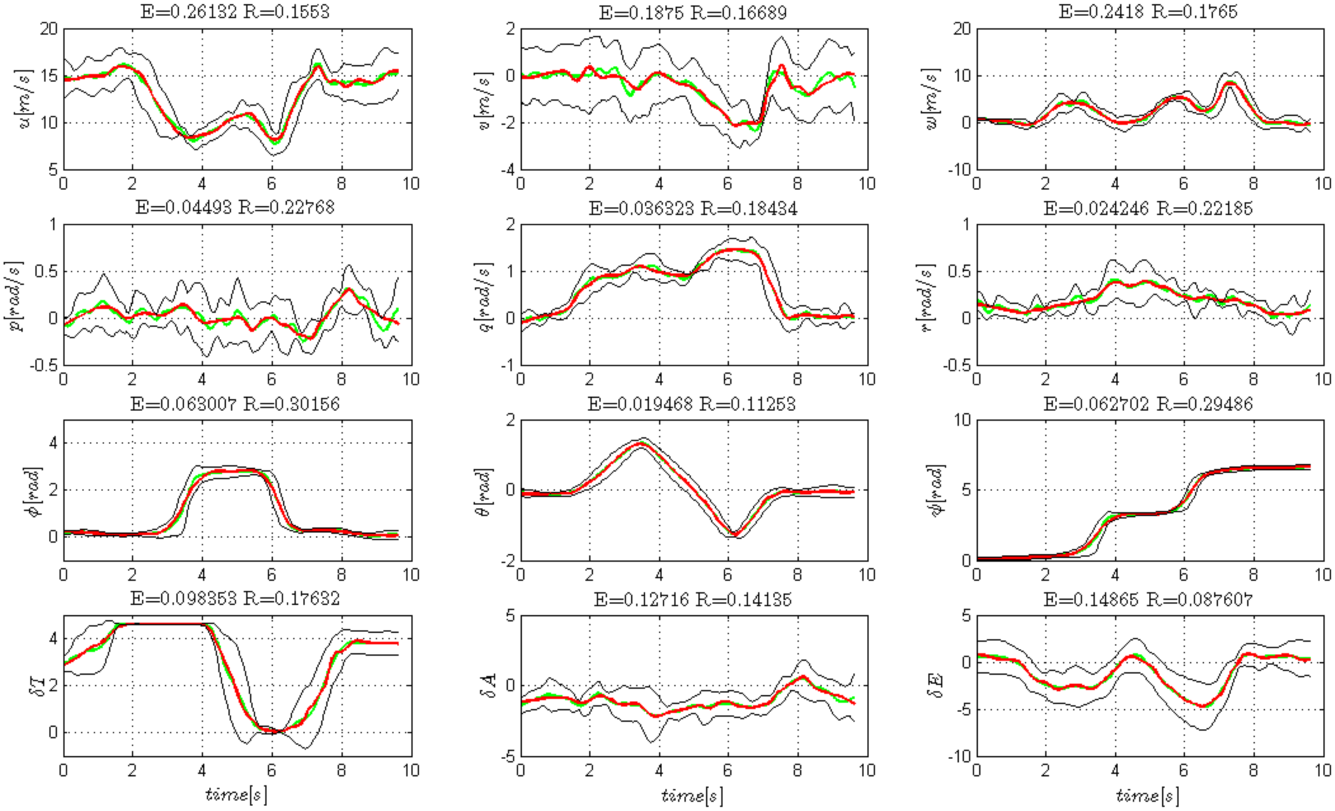

Figure 17: Model performance. In red is the regression signal and in green the mean of the demonstrations

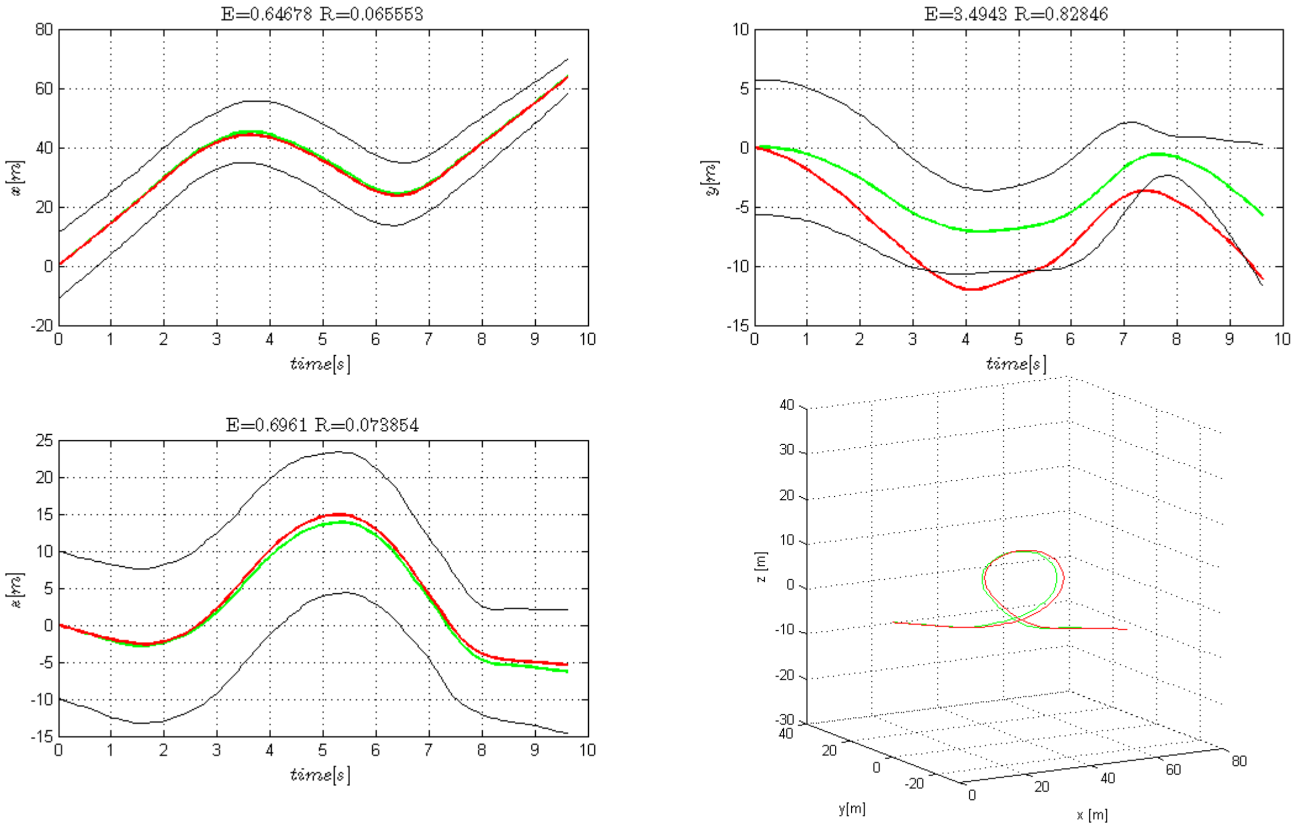

Figure 18: Model performance - trajectory view. In red is the regression signal and in green the mean of the demonstrations

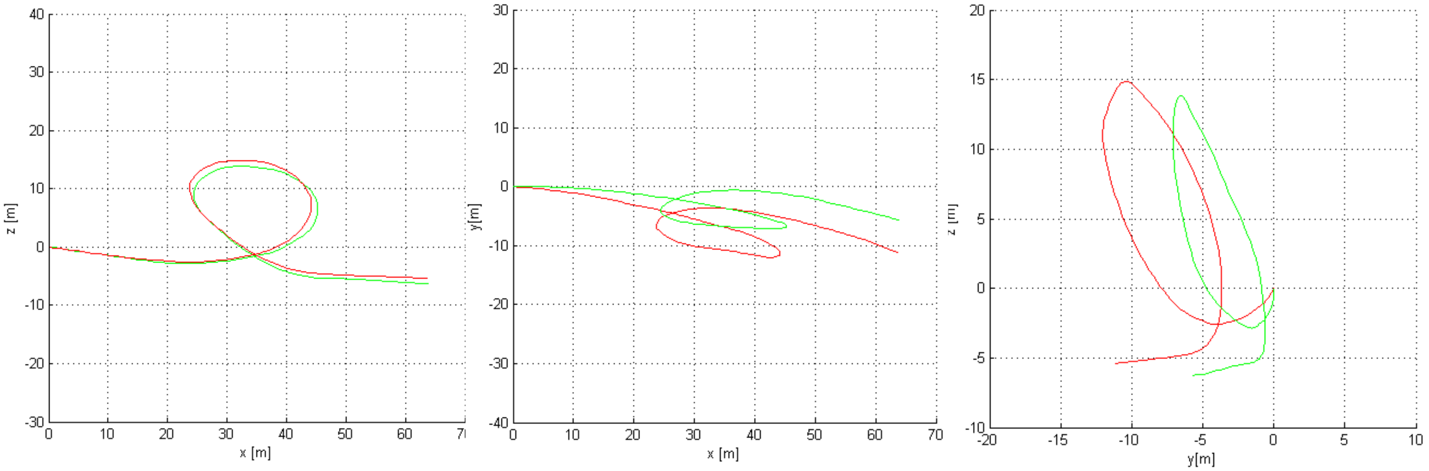

Figure 19: x-z (left), x-y (middle) and y-z (right) view. In red: the regression signal, in green: the mean of the demonstrations

When moving from 35 Gaussian components to just 15, there is, of course, a decrease in performance but it is very small and insignificant when considering by how much the GMM is simplified (Table 1).

Table 1: Performance for D = 13

K	35		15	
	RMSE	R	RMSE	R
u [m/s]	0.2613	0.1552	0.3182	0.1891
v [m/s]	0.1875	0.1668	0.2255	0.2007
w [m/s]	0.2418	0.1764	0.4257	0.3107
p [rad/s]	0.0449	0.2276	0.0591	0.2998
q [rad/s]	0.0363	0.1843	0.0575	0.2919
r [rad/s]	0.0242	0.2218	0.0300	0.2751
φ [rad]	0.0630	0.3015	0.1095	0.5243
ϑ [rad]	0.0194	0.1125	0.0455	0.2631
ψ [rad]	0.0627	0.2948	0.1092	0.5139
δT	0.0983	0.1763	0.1223	0.2193
δA	0.1271	0.1413	0.1755	0.1951
δE	0.1486	0.0876	0.2405	0.1417
x [m]	0.6467	0.0655	0.5990	0.0607
y [m]	3.4942	0.8284	3.4960	0.8288
z [m]	0.6961	0.0738	0.8478	0.0899

In the second case, when the Euler angles are not used in the model, there is an overall loss of performance. Nevertheless, the R ratio remains still very low for both K = 50 and K =15 (Table 2).

Table 2: Performance for D = 10 without Euler angles

K	50		15	
	RMSE	R	RMSE	R
u [m/s]	0.2838	0.1687	0.4020	0.2389
v [m/s]	0.1745	0.1553	0.2363	0.2103
w [m/s]	0.2514	0.1835	0.4135	0.3018
p [rad/s]	0.0461	0.2339	0.0504	0.2557
q [rad/s]	0.0391	0.1987	0.0572	0.2906
r [rad/s]	0.0243	0.2229	0.0259	0.2369
δT	0.0922	0.1654	0.1206	0.2162
δA	0.1384	0.1538	0.1751	0.1946
δE	0.1883	0.1110	0.2780	0.1638

As for the last case, when the Euler angles are used, but not the angular velocities, the overall performance is similar to the previous case (Table 3).

Table 3: Performance for D = 10 without Euler angle rates

K	**40**		**15**	
	RMSE	**R**	**RMSE**	**R**
***u* [m/s]**	0.2347	0.1395	0.3371	0.2003
***v* [m/s]**	0.1900	0.1691	0.2239	0.1993
***w* [m/s]**	0.3034	0.2214	0.4577	0.3340
***φ* [rad]**	0.0390	0.1868	0.1088	0.5210
***ϑ* [rad]**	0.0234	0.1358	0.0456	0.2636
***ψ* [rad]**	0.0386	0.1818	0.1112	0.5230
δT	0.0898	0.1611	0.1285	0.2304
δA	0.1497	0.1664	0.1728	0.1921
δE	0.2016	0.1188	0.3040	0.1791
***x* [m]**	0.5484	0.0555	0.7518	0.0762
***y* [m]**	3.4832	0.8258	3.4028	0.8067
***z* [m]**	0.7113	0.0754	0.8839	0.0937

Choosing the dimension of the model

As was seen in the previously presented results, the best performance seems to be achieved when all the possible dimensions are used and one could argue that using both Euler angles and Euler angle rates would bring a greater reliability to the model.

Are these reasons enough to choose a 13-dimension representation for the demonstration sets? Apart from the non-negligible aspect of computationally expense in case one would want to use the derived model in a control scheme on the MAV, there are other negative aspects to consider.

One of them is the fact that by using both sets of parameters the system might become overdefined. Furthermore, the greater dimensionality also means that more demonstrations would be needed in order to cover adequately the state space and obtain a useful model. This later issue will be better illustrated in the last section of this chapter.

If then one would choose a smaller dimension, which of the two remaining variants should be chosen? Using the (*p*, *q*, *r*) components would mean that all the state parameters would be in the same Aircraft-Body Coordinates System, but this would not bring any real advantage to such a model over the other case.

If one would choose to only use the Euler angles, apart from the dimension reduction, another advantage would be the fact that these parameters, being an integration of the angle rates, contain far less noise.

Using the GMM in a control scheme

The performance analysis previously presented is restricted to a very specific case, when the regression is done with only time as an input and all the other parameters as outputs. This is, of course, the simplest scenario and the good performance is not a surprise.

The regression signal in this case doesn’t use any of the sensor data as input, so that the estimation will not be reactive to the state of the aircraft. Nevertheless, this estimation could still be used as the initialization for an optimal control strategy.

What happens then if more than just time is used as input? This would certainly be the case if the model were to be used for controlling the aircraft.

The figures below show what happens when the sensor data is used, along with time, as input and the commands as output. Only four of the demonstration sets are used for learning, leaving the fifth aside for testing the model. The two cases considered are when using all 13 dimensions and when using 10 dimensions without the Euler angle rates.

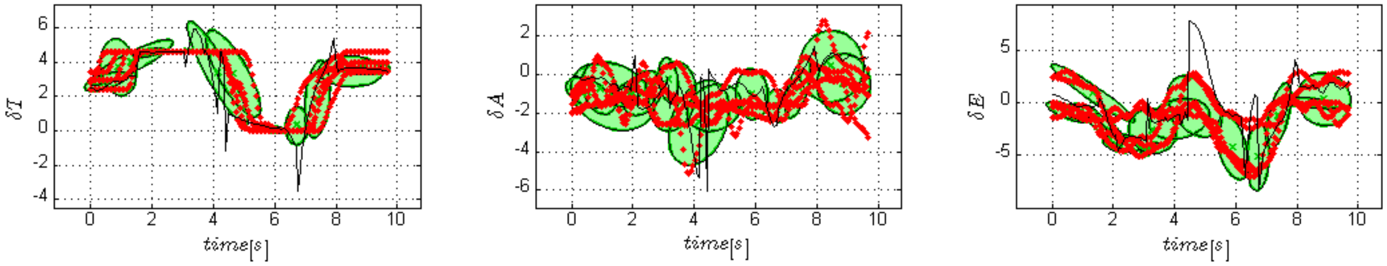

Figure 20: Estimated commands from sensor data as well as time for D = 10 with Euler angles

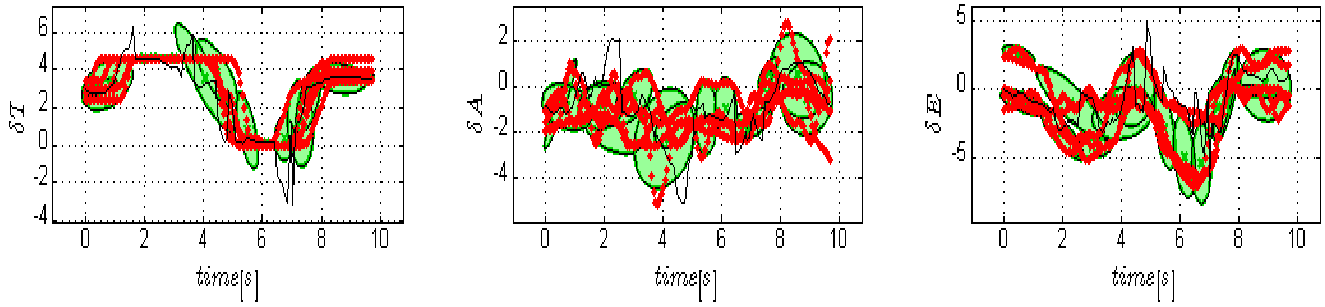

Figure 21: Estimated commands from sensor data as well as time for D = 13

As can be seen, the performance in this case drops drastically. This is, nevertheless, easy to understand because the very low number of demonstration sets used can hardly be expected to cover the high dimensional space of possible trajectories and give a robust model. This idea is enforced by the fact that the results are worse for the 13-dimensional case than for the 10-dimensional one and confirms what was stated in the previous section.

Finally, it must be noted that even if the model would be taught with a high enough number of the demonstration sets as to obtain a good performance for the regressed commands, an even higher number of demonstration sets would be needed to be able to get a regressed signal that would be consistent with the aircraft dynamics.

Conclusion

The purpose of this project was to find a way of using PbD techniques in order to teach an acrobatic maneuver to an aerial robot. The strategy adopted was to first look at ways of representing and aligning the recorded demonstrations data and then extrapolate a model.

After looking at different possibilities the Aircraft-Body Coordinate System was chosen to represent the data, which was subsequently aligned by use of a cost function. Finally, a Gaussian Mixture Model was proposed for modeling the different maneuvers and an analysis of the performance of such a model, together with a reflection on the utility of such a model in a control scheme, were presented.

Due to the high dimensionality of the problem, a direct use of a Gaussian Mixture Model for controlling the aircraft during a maneuver is not obvious and would clearly need that the model be taught with a high number of demonstrations.

I personally feel that this project has been a very useful experience as I have had the opportunity to deal with a complicated dynamical system and learned very much about the way to approach such a problem. It confronted me with the problem of choosing between the different coordinate systems that can be used for representing aeronautical systems in order to best represent the data for Gaussian Mixture Modeling. It also allowed me to get a more hands-on experience with some concepts of Machine Learning, of which I previously had only a theoretical knowledge.

I feel obliged to thank my supervisor, Seyed Mohammad Khansari Zadeh, whose guidance and knowledge of aeronautic systems have been essential in my understanding of this project.

Bibliography

[1] The SMAVNET Project, http://lis.epfl.ch/?content=research/projects/SwarmingMAVs/

[2] S. Leven, J.-C. Zufferey, D. Floreano, A Simple and Robust Fixed-Wing Platform for Outdoor Flying Robot Experiments, EPFL 2008

[3] M. Unser, Image Processing lecture notes, Volume I, 8th edition, EPFL 2008

[4] Wikipedia - ENU Coordinates System, http://en.wikipedia.org/wiki/Geodetic_system

[5] The Mathworks™ - Aerospace Blockset™, About Aerospace Coordinate Systems, http://www.mathworks.com/access/helpdesk/help/toolbox/aeroblks/f3-22568.html

[6] P. H. Zipfel, Modeling and Simulation of Aerospace Vehicle Dynamics, AIAA education series

[7] B. Stevens, F. Lewis, Aircraft Control and Simulation, 2nd Edition. Wiley-Interscience; 2nd edition, October 6, 2003

[8] S. Calinon, F. Guenter, A. Billard, On Learning, Representing and Generalizing a Task in a Humanoid Robot, IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation, No. 5, Vol. 36, 2006

[9] C. M. Bishop, Pattern Recognition and Machine Learning, Springer 2006

Annex

The next graphs show the same strategy that was previously applied to loop maneuvers, in the case of level turns.

For the alignment step the following matrices were used in computing the cost function:

$$Q = diag\left(\frac{1}{\max(|u|)}, \frac{1}{\max(|v|)}, \frac{1}{\max(|w|)}, \frac{1}{\max(|p|)}, \frac{1}{\max(|q|)}, \frac{1}{\max(|r|)}, \frac{4}{\max(|\varphi|)}, \frac{4}{\max(|\vartheta|)}, \frac{4}{\max(|\psi|)}\right)^2$$

$$R = diag\left(0, \frac{1}{\max(|\delta A|)}, 0\right)^2$$

A single example is chosen to illustrate the performance for the level turn case. 50 Gaussian components and all the orientation parameters are used in the model. The number of Gaussians is the one that minimizes the Bayesian Information Criterion (fig. 26). It is, evidently, very large, but it should be noted that the performance remains good also for a lower number of Gaussians as in the previous case.

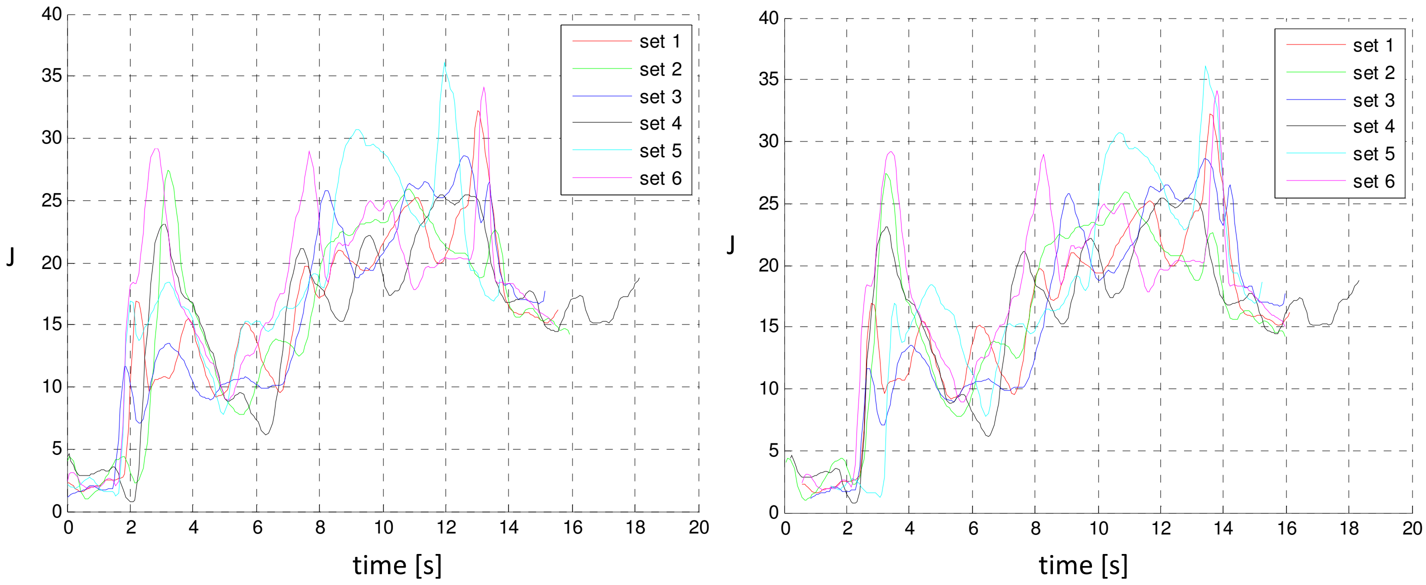

Figure 22: Cost function before (left) and after (right) alignment

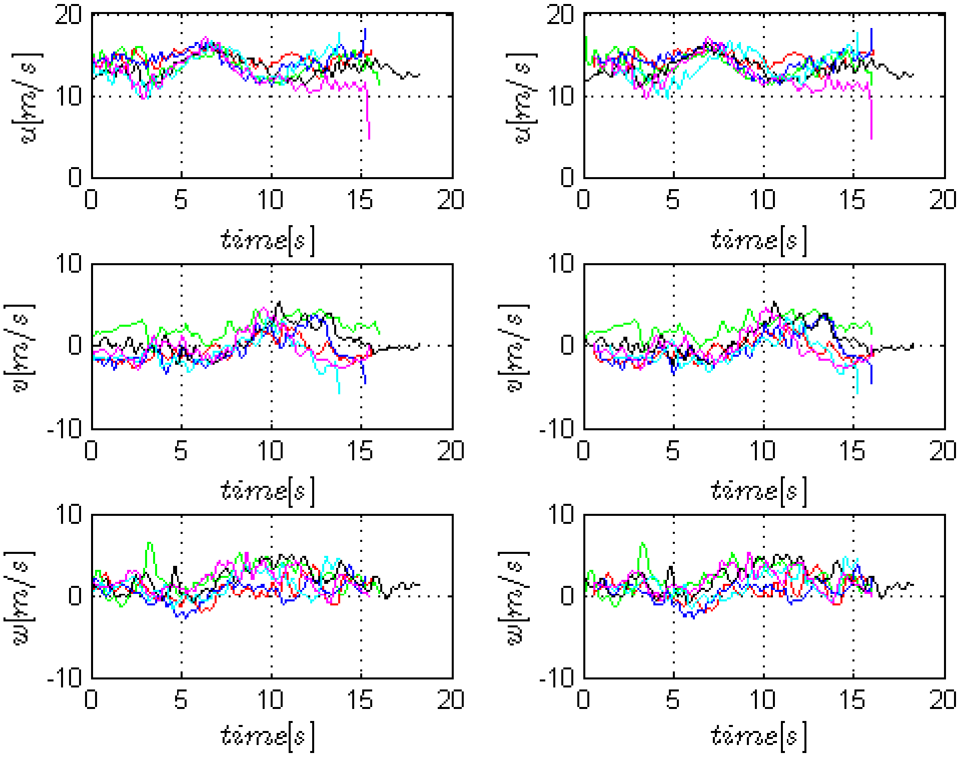

Figure 23: u, v, w before (left) and after (right) alignment

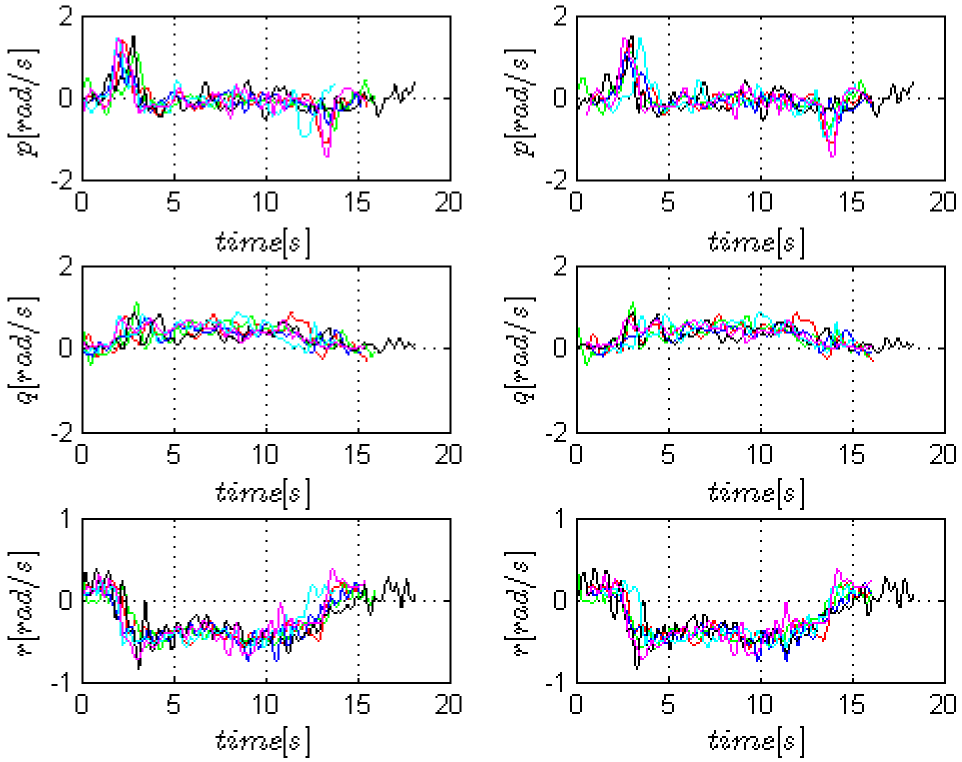

Figure 24: p, q, r before (left) and after (right) alignment

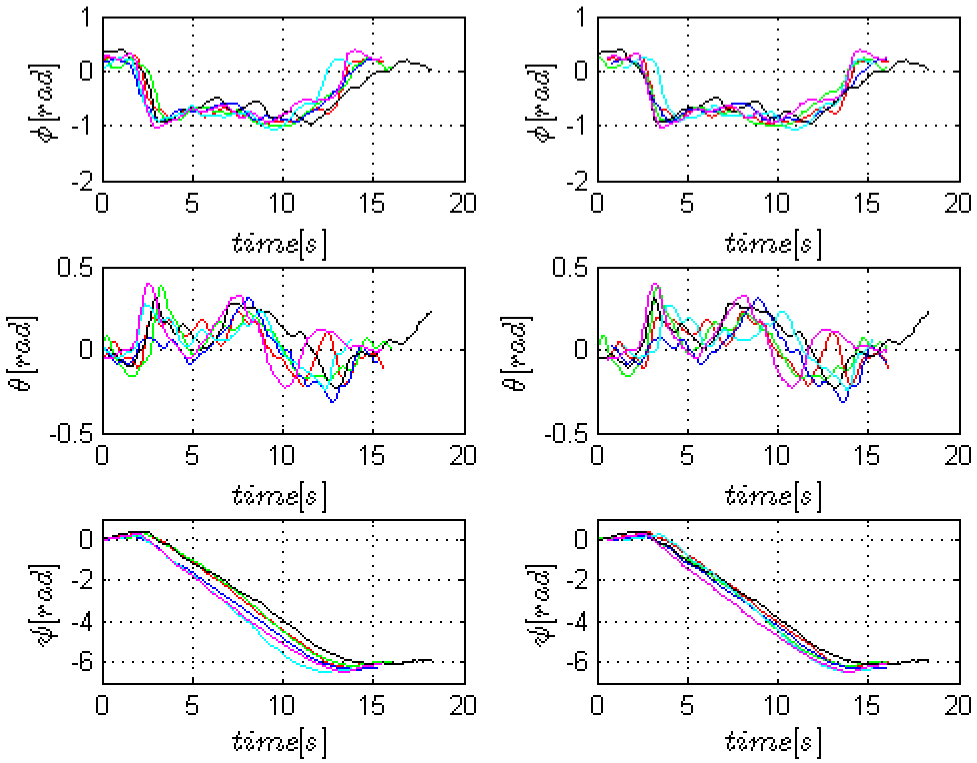

Figure 25: Euler angles before (left) and after (right) alignment

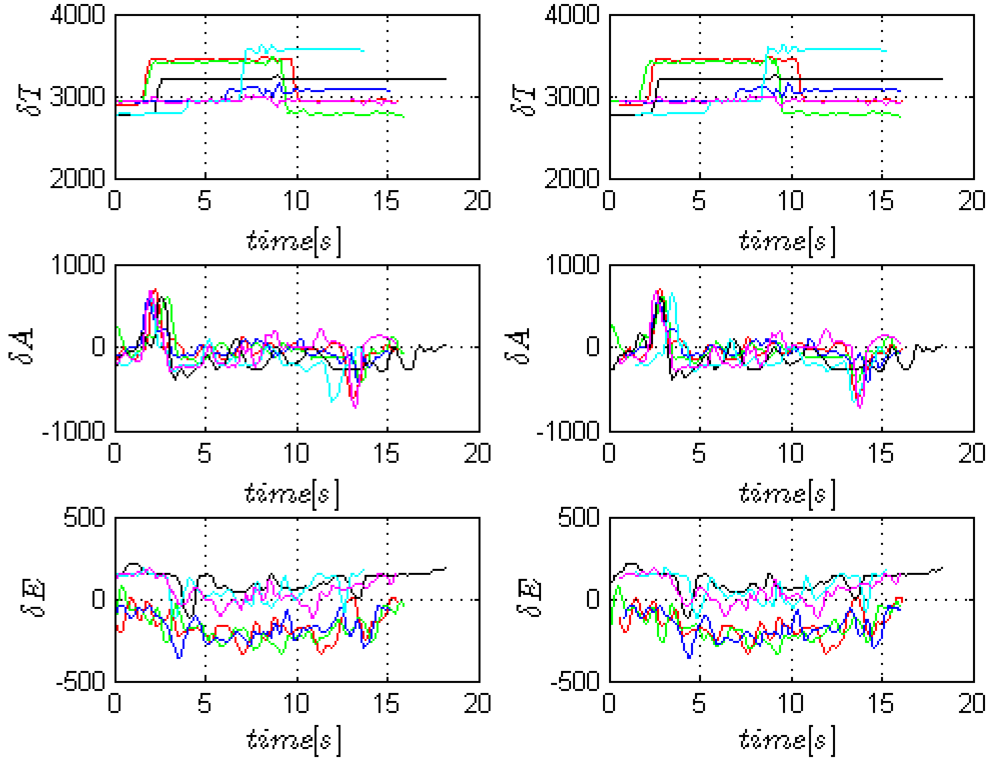

Figure 26: Commands before (left) and after (right) alignment

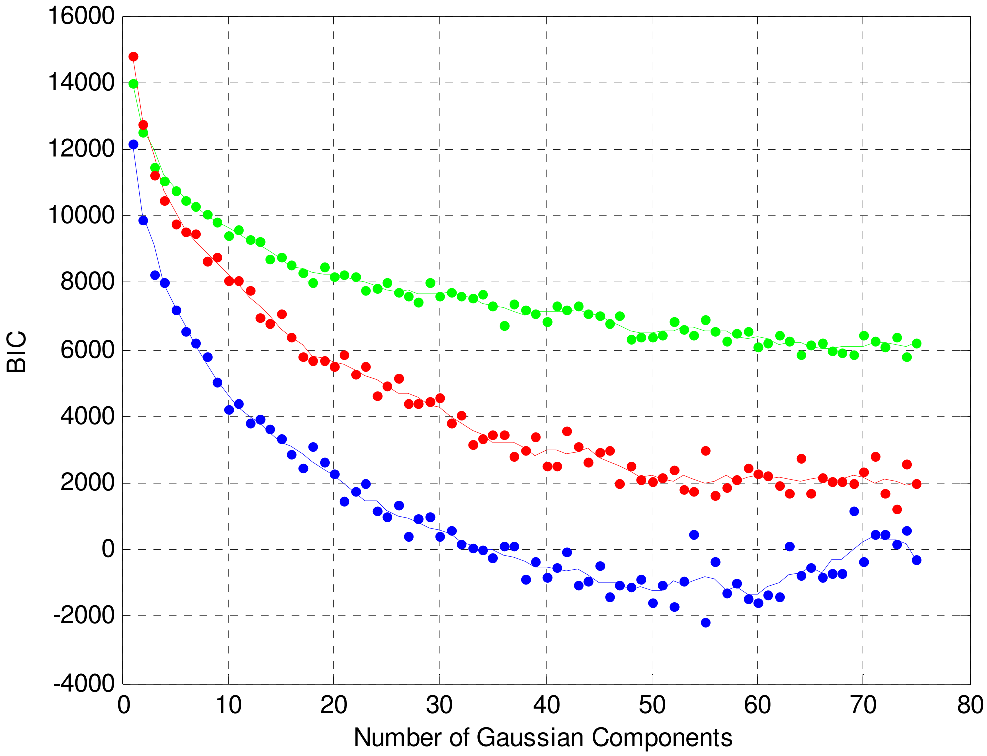

Figure 27: BIC for the case in which all dimensions are used (blue), when the Euler angles are not used (green) and when Euler angle rates are not used (red)

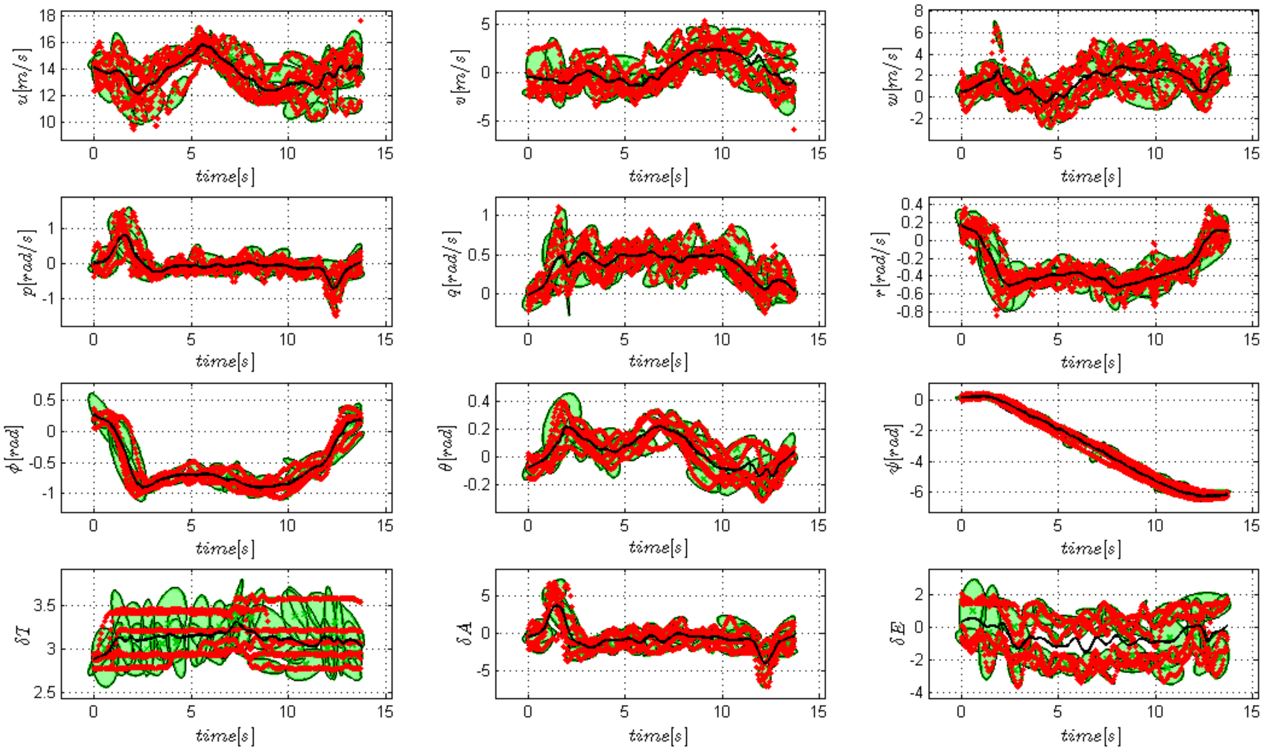

Figure 28: GMM and regression

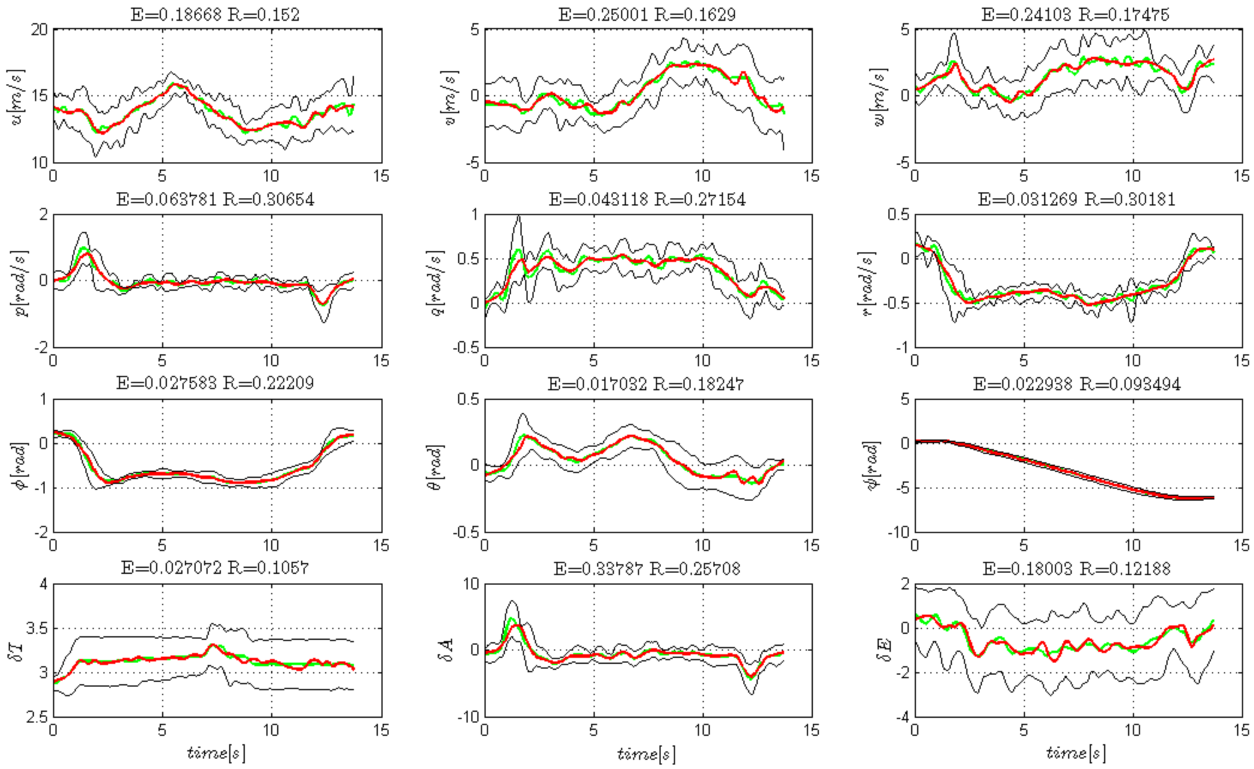

Figure 29: Model performance. In red is the regression signal and in green the mean of the demonstrations

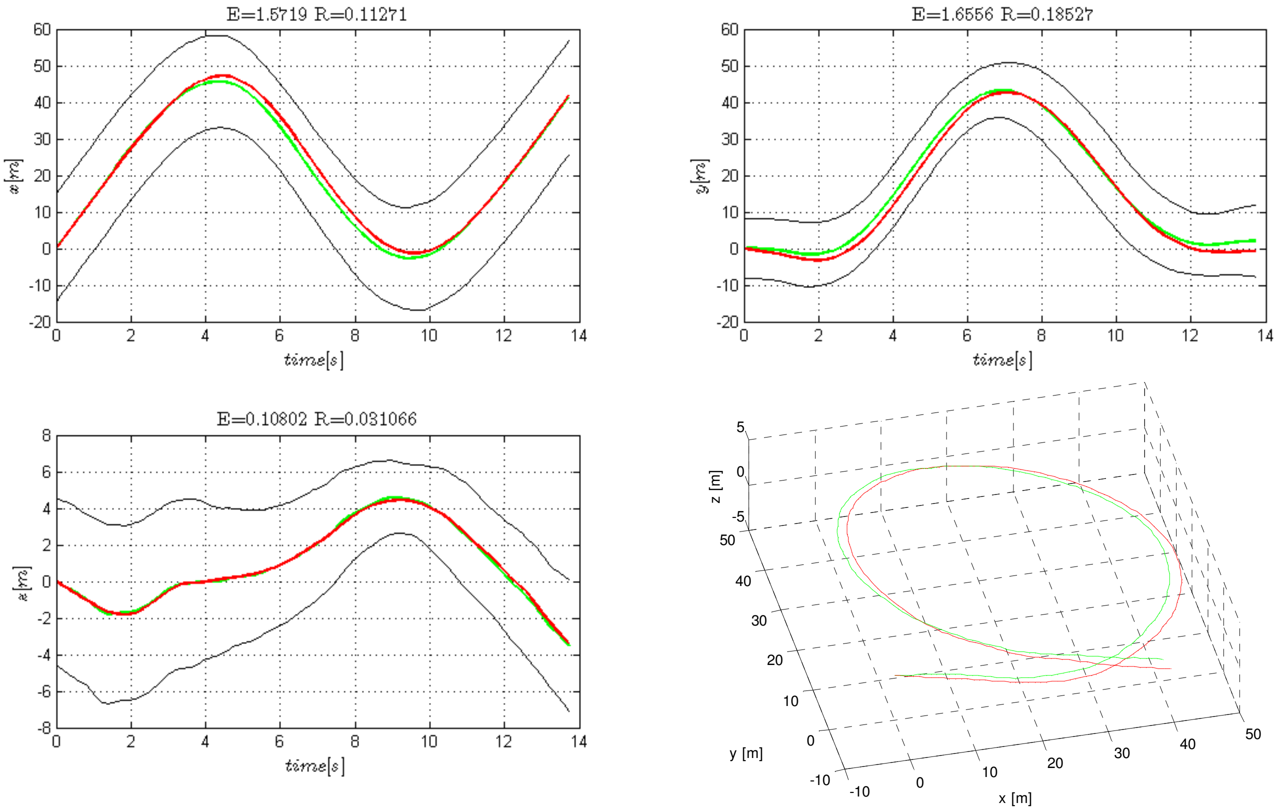

Figure 30: Model performance - trajectory view. In red is the regression signal and in green the mean of the demonstrations