Agile robotic fliers: a morphing based approach

Valentin Riviere, Augustin Manecy, Stéphane Viollet

Agile robotic fliers: a morphing based approach

V. RIVIERE, A. MANECY, S. VIOLLET

April 12, 2018

Abstract

The aerial robot presented here for the first time was based on a quadrotor structure, which is capable of unique morphing performances based on an actuated elastic mechanism. Like birds, which are able to negotiate narrow apertures despite their relatively large wingspan, our Quad-Morphing robot was able to pass through a narrow gap at a high forward speed of $2.5m.s^{-1}$ by swiftly folding up the structure supporting its propellers. A control strategy was developed to deal with the loss of controllability on the roll axis resulting from the folding process, while keeping the robot stable until it has crossed the gap. In addition, a complete recovery procedure was also implemented to stabilize the robot after the unfolding process. A new metric was also used to quantify the gain in terms of the gap crossing ability in comparison with that observed with classical quadrotors with rigid bodies. The performances of these morphing robots are presented, and experiments performed with a real flying robot passing through a small aperture by reducing its wingspan by 48% are described and discussed.

1 Objective

Flying through cluttered environments requires an outstanding level of agility, which often involves the ability to trigger aggressive maneuvers to quickly avoid obstacles or pass through gaps at high speed. In the living world, agility is not restricted to flying insects or even small birds such as hummingbirds. Larger birds such as goshawks [1] and budgerigars [2] are able to negotiate cluttered environments at high speed despite their relatively large wingspan. How do they manage to perform tasks of this kind? By morphing their shape dynamically and reducing their wingspan swiftly by tucking up their wings. Morphing abilities give a flying robot agility by momentarily reducing its wingspan while keeping a sufficiently high payload. Morphing does not require any aggressive maneuvers but fast embedded mechanisms for folding up the robot's structure, as described in [3] for a winged drone.

In the field of robotics, Unmanned Aerial Vehicles (UAVs) are being used increasingly in cluttered and indoor environments for various purposes such as search and rescue expeditions [4], mapping [5] and exploration [6]. The latest flying robots therefore have to be able to avoid collisions and handle narrow gaps successfully. Quadrotors, with their hovering and Vertical Take-Off and Landing (VTOL) abilities, are certainly among the best candidates for meeting these requirements.

Here we focused on designing a narrow gap-crossing strategy which was implemented on a quadrotor. One previous strategy, which has been widely studied consisted in performing aggressive maneuvers to make the quadrotor change its attitude swiftly in order to pass through a vertical or tilted window [7]. Recent studies [8, 9] have succeeded in developing autonomous robotic gap crossing skills based on on-board sensing and computing processes. However, this aggressive attitude control approach has several limitations: the robots have to reach high velocities and angular accelerations which require low inertia of the robot's body as well as high sensor refresh rates, especially in the case of visual sensors so as to prevent blur motion and maintain accurate position estimation with respect to the gap to be crossed.

To address this issue, a new approach was adopted based on morphological changes. Previous authors have presented various types of quadrotors in which the size of the structure can be adapted either passively or dynamically for different purposes. Non-actuated structures were used by [10] to ensure resilience to collision and by [11] to obtain a self-deployable system facilitating the robot's transport. Actuated structures were used by [12] to reduce the wingspan of a hovering robot or to reduce the robot's volume with a scissor-like foldable structure in [13]. Simulated robotic platforms with morphing abilities have been endowed by [14] with full attitude control and by [15] with an interesting tilting rotor mechanism.

Here we present a novel morphing approach whereby a flying quadrotor is endowed with an actuated elastic morphing structure which enables it to cross any gaps encountered at high speed. As shown in the supplementary video, the folded robot is able to pass through apertures that are narrower than the unfolded robot. Section 2 describes in detail the design and the dynamic model on which the morphing robot's structure was based. The control laws and strategy used to stabilize the robot during the folding and unfolding steps are described in section 2.4. The experimental results presented in the last section 3 show the performances of which the Quad-Morphing robot is capable.

Figure 1: (Left) Photo of the Quad-Morphing platform. (Right) Computer-Aided-Design view of the Quad-Morphing platform flying towards a gap while rotating (folding) the arms supporting its four propellers to reduce its wingspan smoothly and quickly so as to avoid colliding with the gap.

2 Material and Methods

2.1 Hardware and Software Overview

As shown in figure 1, we designed and constructed an aerial robotic platform (Quad-Morphing robot) to test the ability of an aerial robot to pass through a gap smaller than its wingspan without any need of aggressive maneuvers. The hardware architecture of the Quad-Morphing robot was based on a previous custom-made platform developed at our lab [16], which consisted of (see figure 2):

- a Gumstix® Overo®'s linux-based Computer-On-Module (COM) for high-level control (attitude/position/folding control) and communication with the ground station via WiFi.
- a NanoWii board carrying a 6-axes Inertial Measurement Unit (IMU MPU6050), which can be used to control manually the quadrotor.
- A very fast and accurate servomotor (MKS DS92A+) actuating the folding system.
- a Rotor Controller board (RCB): a custom-built board giving a low-level control of the rotational speed propellers and providing connections between all the components.

The servomotor implemented on the robot was as fast and light as possible. The folding system was designed so that the two arms could rotate concomitantly to fold or unfold the structure, using a single actuator. Further details about the folding system are given in the following section.

2.2 Mechanical Design and Model

As shown in figure 3, the Quad-Morphing robot was based on a classical quadrotor platform, the mechanical structure of which was greatly adapted to make the robot able to reduce its wingspan dynamically. To simplify the mechanical design, a straightforward approach was used, which consisted in using two pivot links to allow the two arms supporting the robot's propellers to

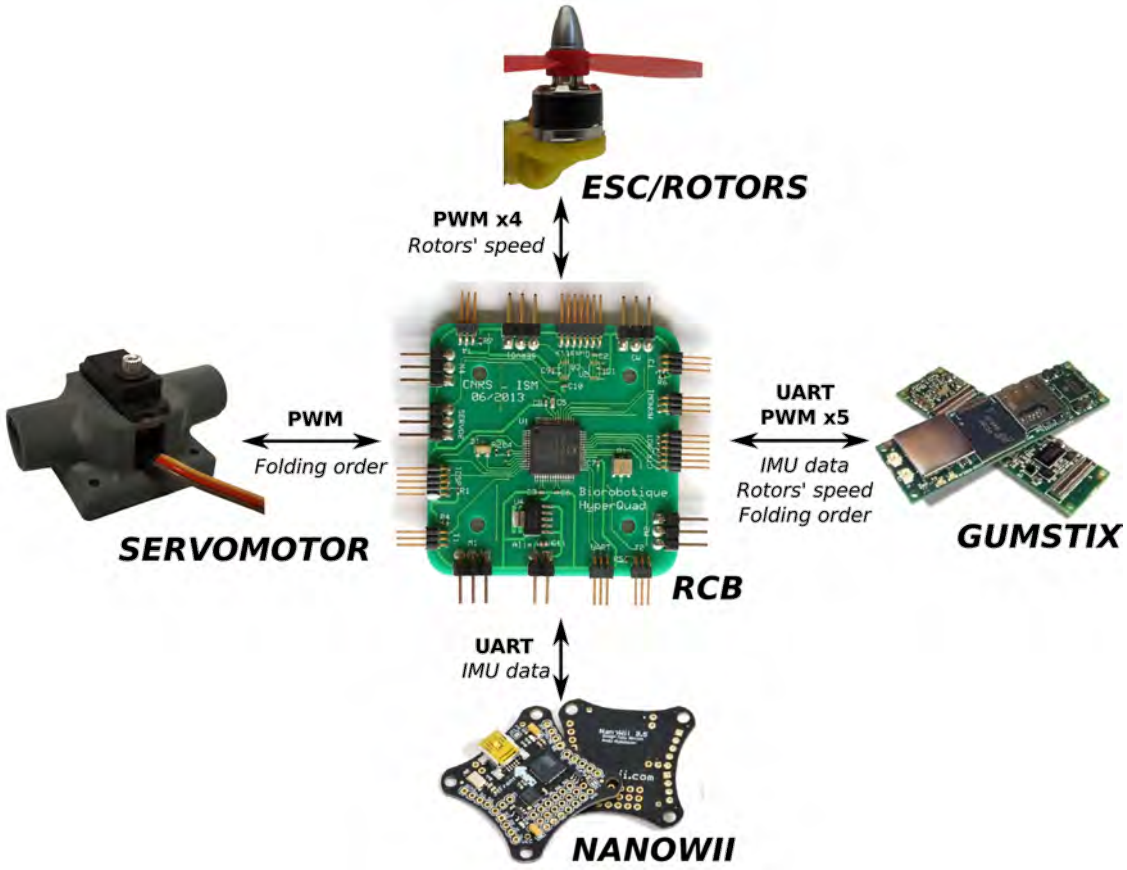


Figure 2: Hardware description: all the signals are transmitted via the custom-made RCB board giving a feedback control of the propeller speeds and voltage levels.

rotate. To keep the robotic platform as light as possible, we implemented a wire-based mechanism composed of two rigid wires fixed to a rotating pulley (in blue in figure 3) mounted onto a fast servomotor and one elastic wire fixed to the edges of the arms and held taut by means of the pulley groove. The tension of the elastic wire was adjusted so as to facilitate the smooth folding and unfolding of the structure and to keep the structure rigid whatever the arms' positions, with no backlash. This mechanism enables the flying robot to fold its arms dynamically back against its body (in figure 3: $\gamma \in [0, 90°]$) and thus to greatly reduce its wingspan to the diameter of a single propeller. Between the two extreme positions: folded ($\gamma = 90°$) and unfolded ($\gamma = 0°$), the robot can reach every intermediate position with a precision which depends on the angular precision of the servomotor (almost 2°).

Size parameters

The Quad-Morphing robot was designed with a $400g$ payload. The body length l_b and arm length l_a (see figure 3) were sized with respect to the propeller's radius $r_{p_{real}}$. During the design phase, we also examined the use of a slightly larger propeller radius ($r_p = 70mm$) than the actual length ($r_{p_{real}} = 64mm$) to prevent the propellers from touching each other. The robot had the following main dimensions, as shown in figure 3:

$$\begin{array}{ll} r_{p_{real}} & = 64mm \\ r_p & = 70mm \\ l_a & = 2.r_p = 140mm \\ l_b & = 4.r_p = 280mm \end{array} \quad (1)$$

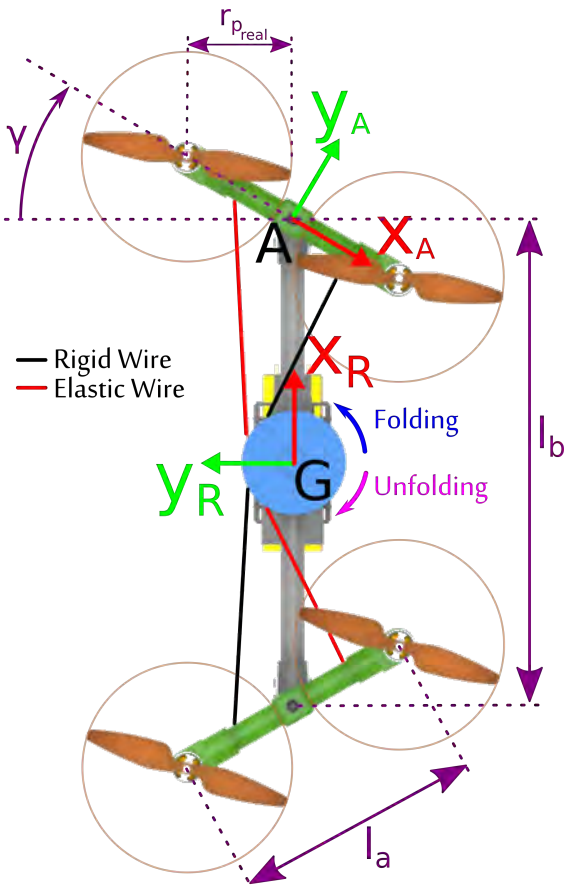


Figure 3: Linkage scheme for the Quad-Morphing robotic platform equipped with an elastic folding mechanism. A servomotor drives the pulley to make the two arms carrying the propellers rotate quickly, resulting in the folding (or unfolding) of the robot's structure. The angle γ, called the folding angle, can range between 0° (unfolded) and 90° (folded).

Dynamic model

To rotate from the inertial frame $\mathcal{I}$ to the robot's frame $\mathcal{R}$ (see figure 1), we used ZYX Euler angles, which corresponds to the rotation matrix $\boldsymbol{R}$ defined as follows:

ψ: Yaw around the Z-axis
θ: Pitch around the Y'-axis
ϕ: Roll around the X"-axis

In the first step, we modeled the Quad-Morphing robot using Newton's equations of motion for dynamic rigid bodies [17], including the thrust and torques applied to the robot, the gyroscopic

effects and the fluid resistance denoted by the coefficient K_v:

$$\begin{aligned}
\dot{\boldsymbol{\xi}} &= \boldsymbol{v} \\
m\dot{\boldsymbol{v}} &= \underbrace{-mg\overrightarrow{\boldsymbol{z_{\mathcal{I}}}} + \boldsymbol{\Gamma}_{(1,:)}.\boldsymbol{\omega}^2\overrightarrow{\boldsymbol{z_{\mathcal{R}}}} - K_v\boldsymbol{v}}_{\text{Sum of forces in } \mathcal{I}} \\
&= -mg\overrightarrow{\boldsymbol{z_{\mathrm{I}}}} + \underbrace{\boldsymbol{R}.\begin{bmatrix} 0 \\ 0 \\ \boldsymbol{\Gamma}_{(1,:)}.\boldsymbol{\omega}^2 \end{bmatrix}}_{\substack{\text{Normal force} \\ \text{produce by propellers}}} - K_v\boldsymbol{v} \\
\dot{\boldsymbol{R}} &= \boldsymbol{R}\boldsymbol{\Omega}_{\times} \\
\boldsymbol{I}_{robot}(\gamma)\dot{\boldsymbol{\Omega}} &= \underbrace{\sum M_G}_{\substack{\text{Sum of} \\ \text{the moments} \\ \text{at G in } \mathcal{R}}} \\
&= \underbrace{-\boldsymbol{\Omega} \times \boldsymbol{I}_{robot}(\gamma)\boldsymbol{\Omega}}_{\text{Gyroscopic term}} + \underbrace{\boldsymbol{\Gamma}_{(2\to4,:)}(\gamma)\boldsymbol{\omega}^2}_{\substack{\text{Torques produce} \\ \text{by propellers}}}
\end{aligned} \tag{2}$$

Where $\dot{\boldsymbol{\xi}} = \boldsymbol{v}$ and $\dot{\boldsymbol{v}}$ are respectively the robot's speed and its acceleration expressed in the inertial frame $\mathcal{I}$, $\boldsymbol{\Omega}_{\times}$ is the skew symmetric matrix of the body's rates, m is the mass, $\boldsymbol{I}_{robot}$ is the inertia matrix, $\boldsymbol{R}$ is the rotation matrix, Γ is the control matrix (see equation 5), where $\boldsymbol{\Gamma}_{(1,:)}$ corresponds to the first line of the matrix Γ, and $\boldsymbol{\omega}$ is the column vector of the propellers' rotational speed. The changes in the inertia $\dot{\boldsymbol{I}}_{robot}$ were neglected since they were taken to be temporary disturbances occurring only during folding maneuvers, i.e., during a very short period (about 200ms, see section 2.3). The inertial accelerations of the arms canceled each other out because they moved in opposite directions during the folding and unfolding processes (see figure 3).

Torques and thrust generated by the quadrotor depend on the square of the propellers' rotational speed, according to the propeller's physics [18], as follows:

$$\begin{aligned}
F_{p_i} &= c_T.\omega^2 \\
\tau_{p_i} &= c_D.\omega^2
\end{aligned} \tag{3}$$

where F_{p_i} and τ_{p_i} are the thrust force and drag moment produced by the i-propeller, respectively. c_T and c_D are the thrust and drag coefficients, respectively.

Torques on the roll and pitch axes also depend on the folding angle γ (see figure 3). The control matrix Γ transforms each propeller's speed into a thrust force T_Σ and moments $\boldsymbol{\tau}_{roll,pitch,yaw}$ as follows:

$$\begin{bmatrix} T_\Sigma \\ \tau_{roll} \\ \tau_{pitch} \\ \tau_{yaw} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{4} F_{p_i} \\ \sum_{i=1}^{4} y_i.F_{p_i} \\ -\sum_{i=1}^{4} x_i.F_{p_i} \\ \sum_{i=1}^{4} \tau_{p_i} \end{bmatrix} = \boldsymbol{\Gamma}(\gamma).\underbrace{\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}}_{\boldsymbol{\omega}^2} \tag{4}$$

Where T_Σ and $\tau_{roll,pitch,yaw}$ are the total thrust and torques applied to the three axes in the robot's frame, respectively. x_i and y_i are the distances on the x-axis and y-axis of the i-propeller. The coefficients of the matrix $\boldsymbol{\Gamma}$ depend on geometrical parameters which are given by:

$$\boldsymbol{\Gamma}(\gamma) = \begin{bmatrix} c_T & c_T & c_T & c_T \\ -c_T.\frac{l_b}{2}.cos(\gamma) & c_T.\frac{l_b}{2}.cos(\gamma) & c_T.\frac{l_b}{2}.cos(\gamma) & -c_T.\frac{l_b}{2}.cos(\gamma) \\ -c_T.(\frac{L}{2} + \frac{l_b}{2}.sin(\gamma)) & -c_T.(\frac{L}{2} - \frac{l_b}{2}.sin(\gamma)) & c_T.(\frac{L}{2} + \frac{l_b}{2}.sin(\gamma)) & c_T.(\frac{L}{2} - \frac{l_b}{2}.sin(\gamma)) \\ c_D & -c_D & c_D & -c_D \end{bmatrix} \tag{5}$$

The coefficients of the matrix Γ are updated in real time to take into account the changes occurring in the folding angle γ due to the rotation of the servomotor (see figure 3). As expected in view of the design of the folding structure, the moment around the roll axis becomes uncontrollable when the folding angle is equal to 90° (see second line of matrix $\boldsymbol{\Gamma}$ in (5)).

Inertia

The robot's inertia ${}_{\mathcal{R}}\boldsymbol{I}_{robot}(G)$ during the folding process, calculated on the basis of the center of mass G with respect to the robot's frame $\mathcal{R}$, depends on the folding angle γ and can be decomposed into the following three matrices: the body's inertia ${}_{\mathcal{R}}\boldsymbol{I}_{body}(G)$ and the two arms' inertia ${}_{\mathcal{R}}\boldsymbol{I}_{arm_1}(A)$ and ${}_{\mathcal{R}}\boldsymbol{I}_{arm_2}(A')$, calculated on the basis of the arms' centers of mass A and A', respectively.

$$ {}_{\mathcal{R}}\boldsymbol{I}_{robot}(G) = {}_{\mathcal{R}}\boldsymbol{I}_{body}(G) + {}_{\mathcal{R}}\boldsymbol{I}_{arm_1}(A) + {}_{\mathcal{R}}\boldsymbol{I}_{arm_2}(A') \tag{6} $$

Computer-Aided-Design software provided the inertia matrix terms in the inertial frame for the body and arms as follows:

$$ {}_{\mathcal{R}}\boldsymbol{I}_{body}(G) = \begin{bmatrix} I_{b_x} & 0 & 0 \\ 0 & I_{b_y} & 0 \\ 0 & 0 & I_{b_z} \end{bmatrix} \; ; \; {}_{\mathcal{A}}\boldsymbol{I}_{arm_1}(A) = {}_{\mathcal{A}}\boldsymbol{I}_{arm_2}(A') = \begin{bmatrix} I_{a_x} & 0 & 0 \\ 0 & I_{a_y} & 0 \\ 0 & 0 & I_{a_z} \end{bmatrix} \tag{7} $$

The formula is calculated for just one arm by obtaining the arm's inertia on the robot's center of mass G in the robot's frame $\mathcal{R}$, using Steiner's theorem and the parallel axis theorem:

$$ \begin{aligned} {}_{\mathcal{R}}\boldsymbol{I}_{arm1}(A) &= \boldsymbol{R}_{\mathcal{A}\to\mathcal{R}} . {}_{\mathcal{A}}\boldsymbol{I}_{arm1}(A) . \boldsymbol{R}^{\boldsymbol{t}}_{\mathcal{A}\to\mathcal{R}} \\ \text{with } \boldsymbol{R}_{\mathcal{A}\to\mathcal{R}} &= \begin{bmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{8} $$

$$ \begin{aligned} {}_{\mathcal{R}}\boldsymbol{I}_{arm1}(G) &= {}_{\mathcal{R}}\boldsymbol{I}_{arm1}(A) + m_{arm1} . {}_{\mathcal{R}}\boldsymbol{D}(\overrightarrow{GA}) \\ \text{with } {}_{\mathcal{R}}\boldsymbol{D}(\overrightarrow{GA}) &= \begin{bmatrix} z_A^2 & 0 & -x_A . z_A \\ 0 & x_A^2 + z_A^2 & 0 \\ -x_A . z_A & 0 & x_A^2 \end{bmatrix} \; ; \; \overrightarrow{GA} = \begin{bmatrix} x_A \\ 0 \\ z_A \end{bmatrix} \end{aligned} \tag{9} $$

Substituting (9) into (6) , we obtain:

$$ {}_{\mathcal{R}}\boldsymbol{I}_{robot}(G) = diag \begin{pmatrix} I_{body_x} + 2.(I_{a_x} . \cos^2(\gamma) + I_{a_y} . \sin^2(\gamma)) + 2.m_{arm} . z_A^2 \\ I_{body_y} + 2.(I_{a_x} . \sin^2(\gamma) + I_{a_y} . \cos^2(\gamma)) + 2.m_{arm} . (x_A^2 + z_A^2) \\ I_{body_z} + I_{a_z} + 2.m_{arm} . x_A^2 \end{pmatrix} \tag{10} $$

As we can see, the inertial matrix ${}_{\mathcal{R}}\boldsymbol{I}_{robot}(G)$ stays diagonal during the folding process, which would not be the case if the arms were rotating in the same direction. We therefore chose the mechanical configuration described in figure 3 with the two arms rotating in opposite directions. It can also be seen from equation (10) that the inertia depends on the folding angle γ on the roll and pitch axes. The controller's gain is therefore automatically adjusted, depending on the state of the robot's structure (folded or unfolded).
From equation (10), the changes in the inertia occurring on each rotational axis due to the folding were calculated:

$$ \begin{aligned} \Delta I_{robot_x} &= -43\% \\ \Delta I_{robot_y} &= +11\% \\ \Delta I_{robot_z} &= 0\% \end{aligned} \tag{11} $$

As expected from the folding mechanism, the inertia on the roll axis decreases dramatically by 43%, whereas the inertia on the pitch axis increases by only 11%. The inertia on the yaw axis remains unchanged. The increase in the pitch inertia by about 11% of its initial value when the robot is unfolded led us to increase the PID controller gain on the pitch axis by this amount accordingly. The decrease in the roll inertia by about 43% reflects the existence of greater instability in the folded position, but the robot's stability was maintained passively by placing the center of mass below the center of thrust in our design.

2.3 Modeling the folding system

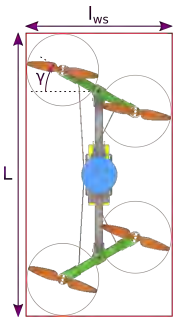
As shown in figure 4, the servomotor regulates the robot's wingspan l_{ws} by adjusting the folding angle γ, as well as by changing the robot's length L in line with the following equations (see figure 3 for definitions of the parameters):

$$ \begin{aligned} L(\gamma) &= l_b + 2.r_{p_{real}} + l_a . sin(\gamma) \\ l_{ws}(\gamma) &= 2.r_{p_{real}} + l_a . cos(\gamma) \end{aligned} \tag{12} $$

A new metric called the folding ratio, defined as the ratio between the folded and unfolded wingspan, was written as follows:

$$\eta = \frac{l_{ws}(\gamma = 90°)}{l_{ws}(\gamma = 0°)} = \frac{2.r_{p_{real}}}{2.r_{p_{real}} + l_a} \tag{13}$$

As shown by equation (13), the gain in terms of the wingspan is directly related to geometric parameters, namely the propeller radius and the length of the arms. In our prototype, the fully folded structure (i.e., $\gamma = 90°$) can adopt a wingspan l_{ws} equal to 128mm, which corresponds to a folding ratio of $\eta = 48\%$, whereas the wingspan of the unfolded structure (i.e., $\gamma = 0°$) is equal to 268mm (see figure 4). In other words, the folded robot is more than 2 times smaller than the unfolded one, which means that the robot can theoretically pass through a gap 2 times smaller than its usual wingspan. The same applies to the robot's length L: the folding ratio is also given in the case of the length in figure 4. Once the wings have folded up, the increase in the robot body's length is 34%.



	l_{ws}	Folding ratio η	L	$\frac{L(\gamma=90°)}{L(\gamma=0°)}$
Unfolded ($\gamma = 0°$)	$268mm$	100%	$408mm$	100%
Folded ($\gamma = 90°$)	$128mm$	48%	$548mm$	134%

Figure 4: Wingspan and various length considerations about wingspan changes depending on the folding angle γ. The folding ratio (see equation (13)) was defined in order to characterize the robot morphing ability in terms of wingspan reduction.

Figure 5 shows experimental curves (8 distinct experiments are presented here) depicting the time course of the folding angle γ and folding ratio η during a folding and unfolding procedure. Data acquisition was made with a Vicon® Sytem motion capture. It can be noted in figure 5 that small bumps appear only during unfolding due to the use of elastic wires in the mechanism (see figure 3). However, the small amplitude of these fast bumps (lasting less than 50ms) make them negligible.
The robot can achieve complete folding, i.e., when the wingspan reaches 95% of its final value, within only $\Delta t_{fold} = 230ms$ ($mean = 230ms, std = 6ms$, *8 experiments*) and complete unfolding within only $\Delta t_{unfold} = 310ms$ ($mean = 310ms, std = 5ms$, *8 experiments*).

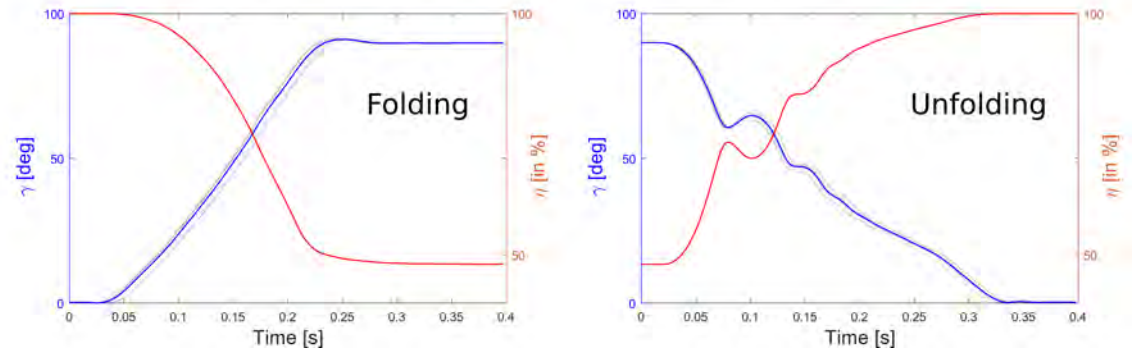


Figure 5: Folding angle (in blue) and folding ratio (in red) versus time

2.4 Gap Crossing Scenario

Here we present a scenario which consists in passing through a gap at high speed. The transversal crossing speed adopted on the X-axis was $2.5m.s^{-1}$, which corresponds to a good trade-off between

a short folding time Δt_{fold} and the limited flight space available for our experiments. In all these experiments, the width of the gap was smaller than the unfolded robot's wingspan, which caused the robot to adopt the folded configuration, as shown in the frames extracted from a video-recorded gap crossing test (see figure 6).
This scenario can involve three different modes, the activation of which depends on the position of the robot with respect to the gap (see figure 7):

- **Full control mode**: The attitude and position controllers are fully activated and run in real time onboard the robot. Thanks to the trajectory planner described in figure 9, we can generate a feasible trajectory in terms of the acceleration and the speed in order to make the robot reach the speed V_{A_x} as fast as $2.5m.s^{-1}$ in the steady state when crossing the point A (see figure 7).

- **Degraded control mode**: immediately after the folding of the two arms, the integrator output signal (see equation (15)) delivered by the roll attitude controller is held constant. The position of point A prior to the gap is determined by applying the following equation: $\Delta X_{fold} = V_{A_x}.\Delta t_{fold}$, where V_{A_x} and Δt_{fold} are the robot's speed at point A and the folding time, respectively.

- **Recovery control mode**: once the gap has been passed, a recovery mode procedure is initiated in order to re-stabilize the robot at point B, which is defined by a position with respect to the position of the gap: $\Delta X_{unfold} = \frac{L(\gamma=90^\circ)}{2}$.

We also implemented a supervisor which makes it possible to switch sequentially between the three modes described above so as to adjust the gains and activate the various controllers as required, depending on the robot's position along the trajectory imposed by the planner.

2.5 Control and State Estimation

All the control laws, the estimator and the trajectory planner described in figure 8, are implemented on the Gumstix® computer on module via the RTMaG toolbox [19] developed at our laboratory and with QUARC® running in the MATLAB® environment in the ground station with WiFi communications. Position estimation and yaw measurements are provided by off-board sensors: a Vicon® system of localization was used to determine the robot's position in real time and to emulate a magnetometer, which is not implemented on our platform. The motion capture system (the VICON system) can locate with a great accuracy [16] the Quad-Morphing robot in our flight arena, which is equipped with 17 cameras, with great accuracy.
Quadrotor robots are not fully actuated systems: four inputs corresponding to the four propeller's speeds control six outputs, which are the six degrees of freedom of the robot's rigid body (attitude and position in 3D). To control systems of this kind, we used two cascaded controllers (as described in figure 8): one position controller providing the thrust and two accelerations in the horizontal plane, and one attitude controller which receives horizontal accelerations and yaw angles as inputs and delivers outputs consisting of the three torques (roll, pitch and yaw). The motor controller then determines the propellers' speeds by means of the control matrix Γ, the inputs of which are the thrust and the three torques.

Position and yaw estimation

The 3-D position and the heading are the only measurements acquired by the motion capture system at a sampling frequency of 500Hz. The data are then filtered by a Kalman filter to reduce the noise and sent via WiFi at a frequency of 200Hz to the Gumstix® computer embedded on board.

Attitude estimation

A complementary filter [20] was added for estimating the robot's attitude and de-biasing the rate gyros. Unfortunately the IMU does not include any magnetometers for estimating the heading, which was specified in real time by the ground station via WiFi.

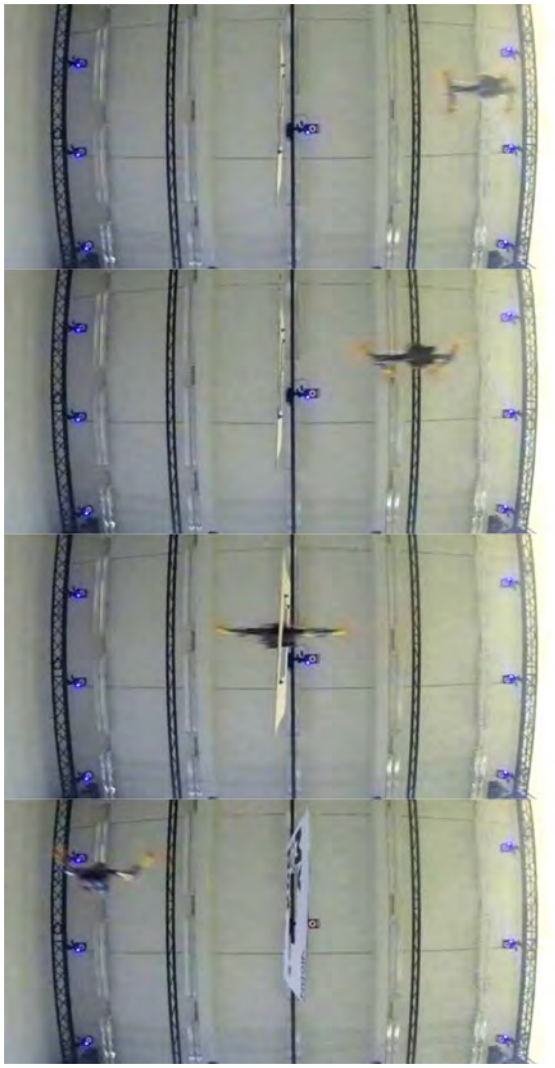


Figure 6: Bottom view: sequence of the Quad-Morphing platform passing through a gap.

With the gyro and accelerometer measurements $\bar{\boldsymbol{\Omega}}$ and $\overrightarrow{\boldsymbol{g}}_{\boldsymbol{imu}}$, respectively, we obtain:

$$\begin{cases} \dot{\hat{q}} = \frac{1}{2}\hat{q} \otimes \boldsymbol{p}(\bar{\boldsymbol{\Omega}} - \hat{\boldsymbol{b}} - \boldsymbol{\alpha}) \\ \dot{\hat{\boldsymbol{b}}} = \boldsymbol{k_b}.\boldsymbol{\alpha} \\ \boldsymbol{\alpha} = \boldsymbol{k_a} \circ \frac{\overrightarrow{\boldsymbol{g}} \times (\hat{\boldsymbol{q}} \odot \overrightarrow{\boldsymbol{g}}_{imu})}{\|\overrightarrow{\boldsymbol{g}}\|^2} + \boldsymbol{k_v} \odot \tilde{s}.\tilde{\boldsymbol{v}} \end{cases} \tag{14}$$

Where $\otimes$ is the Hamilton product, $\boldsymbol{p}(\boldsymbol{v})$ is defined by $\boldsymbol{p}(\boldsymbol{v}) = \begin{pmatrix} 0 & \boldsymbol{v} \end{pmatrix}^t$, $\circ$ is the Hadamard product, and $\odot$ is the quaternion-vector product (defined by $\boldsymbol{q} \odot \boldsymbol{v} = \boldsymbol{q} \begin{bmatrix} 0 & \boldsymbol{v} \end{bmatrix}^t \boldsymbol{q}^{-1}$). The quaternion error is defined by $\tilde{\boldsymbol{q}} = \begin{pmatrix} \tilde{s} & \tilde{\boldsymbol{v}} \end{pmatrix}^t = \bar{\boldsymbol{q}}_{vicon}^{-1} \otimes \hat{\boldsymbol{q}}$, $\tilde{s}$ and $\tilde{\boldsymbol{v}}$ are the scalar part and the vectorial part of the error respectively, $\boldsymbol{k_a} = \begin{pmatrix} k_{a1} & k_{a2} & 0 \end{pmatrix}$ and $\boldsymbol{k_v} = \begin{pmatrix} 0 & 0 & k_{v3} \end{pmatrix}$ are weighting matrices which enable us to use the accelerometer alone to determine the roll and pitch values and the external Vicon® to determine the yaw value.

Attitude and Position Controllers

The quaternion attitude controller implemented on-board based on [21] [22], ensures overall asymptotic stability. The following control law was adopted:

$$\begin{aligned} \boldsymbol{\Omega_{des}} &= sign(\tilde{s}_{eq}).\boldsymbol{K_\Omega}.\tilde{\boldsymbol{v}}_{\boldsymbol{eq}} \\ \boldsymbol{\tau_{des}} &= I.\left(\boldsymbol{k_\tau}.(\boldsymbol{\Omega_{des}} - \hat{\boldsymbol{\Omega}}) + \boldsymbol{k_{int}} \int_0^t \boldsymbol{e_q}.dt\right) \end{aligned} \tag{15}$$

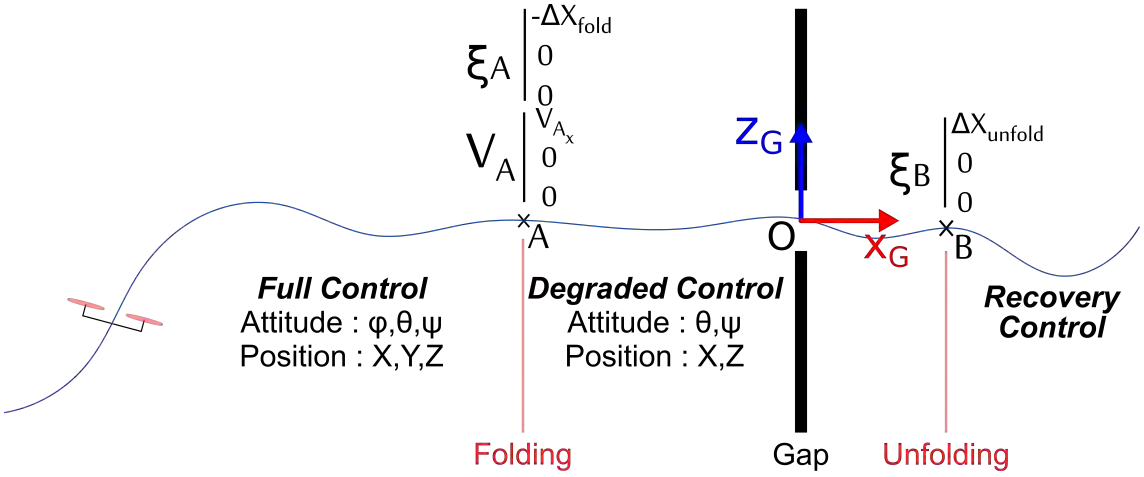


Figure 7: Side view of the gap crossing scenario including the three modes which were activated sequentially, depending on the robot's position along its trajectory: 1) Full Control before reaching point A, 2) Degraded Control from point A to B, 3) Recovery Control after point B

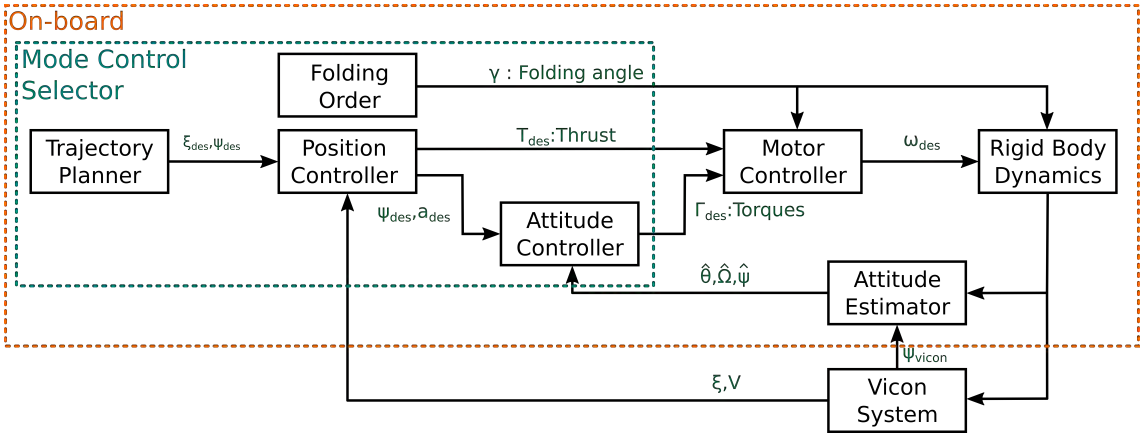


Figure 8: Block diagram of the Quad-Morphing robot's control system.

Where $\tilde{\boldsymbol{q}}_{eq}$ and $\boldsymbol{e_q}$ are the quaternion and the vector error, respectively, defined by:

$$\begin{aligned} \boldsymbol{e_q} &= \tilde{s}_{eq}.\tilde{\boldsymbol{v}}_{eq} \\ \tilde{\boldsymbol{q}}_{eq} &= \begin{pmatrix} \tilde{s}_{eq} & \tilde{\boldsymbol{v}}_{eq} \end{pmatrix}^t = \hat{\boldsymbol{q}}^{-1} \otimes \boldsymbol{q}_{des} \end{aligned} \tag{16}$$

The attitude required is given directly by XY-Axis accelerations and thrust adjustments imposed by the position controller:

$$\begin{cases} \phi_{des} = \sin^{-1}\left(\frac{m}{T_{des}}(a_{1,des}\sin\hat{\psi} - a_{2,des}\cos\hat{\psi})\right) \\ \theta_{des} = \sin^{-1}\left(\frac{m}{T_{des}}(a_{1,des}\cos\hat{\psi} + a_{2,des}\sin\hat{\psi})\right) \end{cases} \tag{17}$$

The position controller implemented was a proportional-integral-derivative (PID) controller defined as follows:

$$\boldsymbol{a_{FeedBack}} = \boldsymbol{k_p}(\boldsymbol{\xi_{des}} - \bar{\boldsymbol{\xi}}) + \boldsymbol{k_i}\int_0^t (\boldsymbol{\xi_{des}} - \bar{\boldsymbol{\xi}})dt + \boldsymbol{k_d}\left(\frac{d\boldsymbol{\xi}_{des}}{dt} - \bar{\boldsymbol{v}}\right) \tag{18}$$

By adding a feed-forward term, the required thrust T_{des} and acceleration $a_{1:2,des}$ were directly obtained on the X and Y axes as follows:

$$\begin{cases} T_{des} = \frac{m}{\cos\hat{\phi}\cos\hat{\theta}}\left(a_{3,FeedBack} + g + \frac{d^2\xi_{3,des}}{dt^2}\right) \\ a_{1:2,des} = a_{1:2,FeedBack} + \frac{d^2\boldsymbol{\xi}_{1:2,des}}{dt^2} \end{cases} \tag{19}$$

Numerical values of the controller coefficients are given by:

	k_Ω	k_Γ	k_{int}
Roll	4	13	0
Pitch	5	10	0
Yaw	9	3	10

	k_p	k_d	k_i
X	0	3	0
Y	8	6	2
Z	11	3	1

There is no direct adaptation term in the PID controller. The control adaptation is made by updating the coefficients of the control matrix Γ as described in 2.2.

Trajectory planning

As described in figure 9, a pre-filter was added to generate speeds and accelerations which are compatible with the robot's dynamics. This trajectory planner consists of three cascaded integrators ensuring smooth accelerations, which are injected into the feed-forward position controller. Physical constraints (speed and acceleration limits) were also added to each integrator by specifying saturation values giving feasible trajectories. This trajectory generator features a good trade-off between computational ressources and physical constraints requirements (more complex trajectory generator could have been used as [23] or [24]).

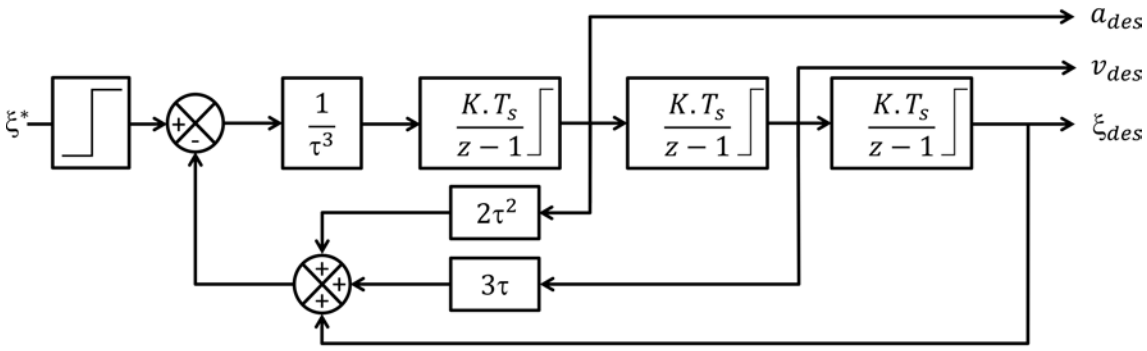


Figure 9: The trajectory planner generates accelerations, speeds and positions compatible with feasible trajectory. The integrators are presented here in their discrete forward-Euler form, where T_s is the sampling time, which is equal to $10ms$, $K = 1$ and $\tau = 500ms$.

2.6 Recovery Control

To achieve complete stabilization of the robot once the gap has been crossed, we implemented a recovery control procedure consisting of a control state machine inspired by [25].
The state mechanism involves four sequentially event-triggered steps depending on the conditions defined as follows:

1. **Step 1: attitude stabilization**: only the attitude controller is turned on with $(0, 0, 0)$ as the required attitude on the three rotational axes (roll, pitch and yaw) and the thrust are kept constant at $T = m.g$ to compensate for gravity. Step 2 is carried out once the following condition 1 has been reached:

$$\textbf{CONDITION 1:} \begin{cases} |\theta| < 20° \\ |\phi| < 20° \\ |\dot{\theta}| < 10 rad.s^{-1} \\ |\dot{\phi}| < 10 rad.s^{-1} \end{cases} \tag{20}$$

2. **Step 2: vertical speed control**: the feedback control of the robot's vertical speed is turned on with the required speed equal to zero. When the speed conditions defined below in (21) are satisfied, the required vertical position z_{des} becomes equal to the estimated vertical position $\hat{z}$ just before switching to step 3.

$$\textbf{CONDITION 2:} \quad |\dot{z}| < 0.3 m.s^{-1} \tag{21}$$

$$\Rightarrow z_{des} = \hat{z}$$

3. **Step 3: Longitudinal and lateral speed control**: activation of the horizontal speed control with the required speed equal to zero and the vertical position control with a new required vertical position. When condition 3 relating to the X and Y axes (22) is satisfied, the position controller locks the actual estimated positions $\hat{x}, \hat{y}$ before switching to step 4.

$$\textbf{CONDITION 3: } \begin{cases} |v_x| < 0.5m.s^{-1} \\ |v_y| < 0.5m.s^{-1} \end{cases} \Rightarrow \begin{cases} x_{des} = \hat{x} \\ y_{des} = \hat{y} \end{cases} \tag{22}$$

4. **Step 4: full control**: activation of the attitude and position control system with new required positions $\boldsymbol{\xi}_{des} = (x_{des}, y_{des}, z_{des})^t$.

2.7 Metrics and calibration

Metrics

As the Quad-Morphing robot is the first prototype showing both morphing and gap crossing abilities, some new metrics were adopted in order to show the robot's performances, especially their repeatability. The main metric adopted for this purpose was the projected wingspan on the gap plane on the Y-axis, called w_{proj} (see figure 10), which was defined as follows:

$$w_{proj}(\gamma, \psi) = \frac{l_{ws}(\gamma)}{\cos\psi} \tag{23}$$

When the robot is folded, with (12):

$$\begin{array}{ll} L & = L(\gamma = 90°) = l_b + l_a + 2.r_{p_{real}} \\ l_{ws} & = l_{ws}(\gamma = 90°) = 2.r_{p_{real}} \end{array} \tag{24}$$

Which leads us to avoid colliding with the gap, with the width of which is denoted w_{gap}:

$$w_{proj}(\psi) = \frac{2.r_{p_{real}}}{\cos\psi} < w_{gap} \tag{25}$$

Likewise on the Z-axis in relation with the robot's height h and the elevation angle ϵ, which corresponds to the angle between x_R, the robot's axes and x_G, the gap's axes on the (Ox_Gz_G) plane, with the height of the gap defined by h_{gap}:

$$h_{proj}(\epsilon) = \frac{h}{\cos\epsilon} < h_{gap} \tag{26}$$

with:

$$\epsilon = -\tan^{-1}\left(\frac{\tan\theta}{\cos\psi}\right) \tag{27}$$

As shown in figure 10, we introduced the two points denoted w_{left} and w_{right} which correspond to the left and right edges of the cross section between the gap and the robot's span on the Y-axis (h_{up} and h_{down} for the top and bottom edges, respectively, on the Z-axis):

$$\begin{array}{ll} w_{left} & = \Delta Y + \frac{w_{proj}}{2} - \Delta X.\tan\psi \\ w_{right} & = \Delta Y - \frac{w_{proj}}{2} - \Delta X.\tan\psi \\ h_{up} & = \Delta Z + \frac{h_{proj}}{2} - \Delta X.\tan\epsilon \\ h_{down} & = \Delta Z - \frac{h_{proj}}{2} - \Delta X.\tan\epsilon \end{array} \tag{28}$$

(28) shows the importance of having an efficient and accurate ψ-angle control, but this can also be said in the case of the Y-axis and Z-axis control in order to minimize ΔY and ΔZ, i.e., the distance between the robot's center of mass and the geometrical center of the gap on the Y-axis and the Z-axis in the gap's frame $\mathcal{G}$, as can be seen in figure 10. We also considered the case where

the robot is outside the gap: this situation requires new definitions of the various points of interest, as follows (see figure 10):

$$\begin{array}{ll} & \begin{cases} w_{left} = \Delta Y + \frac{L(\gamma)}{2}\sin\psi + \frac{l_{ws}(\gamma)}{2}\cos\psi \\ w_{right} = \Delta Y + \frac{L(\gamma)}{2}\sin\psi - \frac{w_{proj}(\gamma,\psi)}{2} \end{cases} \\ otherwise & \begin{cases} w_{left} = \Delta Y + \frac{L(\gamma)}{2}\sin\psi + \frac{w_{proj}(\gamma,\psi)}{2} \\ w_{right} = \Delta Y + \frac{L(\gamma)}{2}\sin\psi - \frac{l_{ws}(\gamma)}{2}\cos\psi \end{cases} \\ if\ \epsilon \geq 0 & \begin{cases} h_{up} = \Delta Z + \frac{L(\gamma)}{2}\sin\epsilon + \frac{h}{2}\cos\epsilon \\ h_{down} = \Delta Z + \frac{L(\gamma)}{2}\sin\psi - \frac{h_{proj}(\gamma,\epsilon)}{2} \end{cases} \\ otherwise & \begin{cases} h_{up} = \Delta Z + \frac{L(\gamma)}{2}\sin\epsilon + \frac{h_{proj}(\gamma,\epsilon)}{2} \\ h_{down} = \Delta Z + \frac{L(\gamma)}{2}\sin\psi - \frac{h}{2}\cos\epsilon \end{cases} \end{array} \quad (29)$$

Calibration

To improve the robot's performances, several experiments were conducted with the same gap-crossing speed $V_{A_X} = 2.5m.s^{-1}$ in order to compensate for constant perturbations on the Y-axis and the Z-axis during folding by biasing (ΔY_{bias} and ΔZ_{bias}) the robot's position with respect to the center of the gap. The biases were determined by finding the middle of the most extreme trajectories during our experiments when the robot's center of mass was on the same plane as the gap:

$$\text{for } \Delta X = 0\text{:} \quad \begin{cases} \Delta Y_{bias} = \frac{max(w_{left}) + min(w_{right})}{2} \\ \Delta Z_{bias} = \frac{max(w_{up}) + min(w_{down})}{2} \end{cases} \quad (30)$$

Therefore to compensate for these biases, the required positions in the gap frame $\mathcal{G}$ during the approach now become:

$$\begin{aligned} Y_{des} &= -\Delta Y_{bias} \\ Z_{des} &= -\Delta Z_{bias} \end{aligned} \quad (31)$$

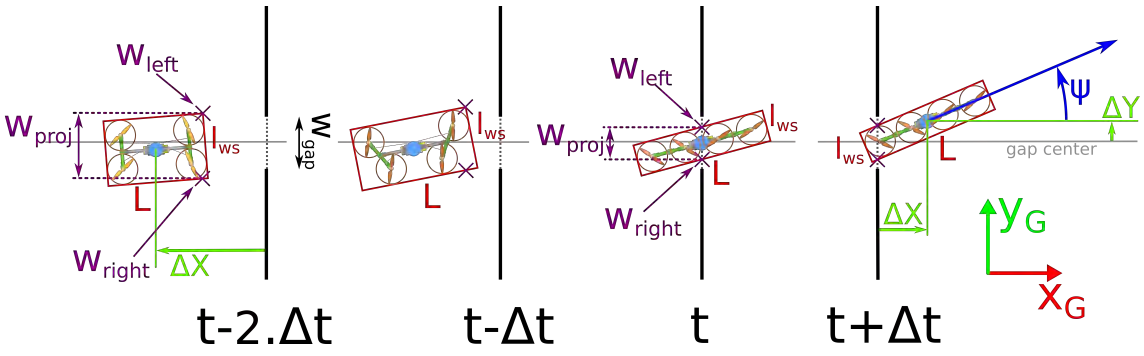


Figure 10: Top view during gap crossing with w_{left} and w_{right} and metric w_{proj}

3 Results

3.1 Experimental Setup

The Vicon® motion system was used instead of an embedded magnetometer and to determine the robot's position in real time. Our RT-MaG custom toolbox [19] was used to run the Simulink model on our embedded Linux computer module and to monitor the data in real time with Quanser® Real-Time Control (QUARC®) running on the ground station. A video of the experimental setup is provided in the attached files.

3.2 Crossing the gap

In the eight experiments presented here, we can see the robot's extreme positions during the final approach and the crossing of the gap ($20 \times 20cm^2$). All the positions are given in the gap frame $\mathcal{G}$, where the origin is taken to be the center of the gap.

As shown in figure 11, the robot was able to perform 8 consecutive gap crossing trials without colliding or touching the sides of the gap. Figure 11 shows edges w_{left} and w_{right} (as defined in figure 10) in the case of each experiment versus the robot's position on the X_G-axis and h_{up}, h_{down} gives the positions of the edges of the robot on the (Ox_Gz_G) plane. The area between the two blue dashed vertical lines indicates possible positions where the robot is liable to collide with the gap (the gap's width and height are presented in horizontal solid red lines in the figures) and the X-axis corresponds to the robot's center of mass on the X_G-axis. In the figure 11, we can see the occurrence of a loss of altitude due to the lower available thrust in the folded position. Once the robot's arms have been folded, the two rotors placed just above the robot's body lead to an overlapping effect which is responsible for the loss of propeller lift. The resulting loss of altitude is not totally compensated for by the controller, which is subjected to thrust limitations in order to prevent the motors from being saturated and to keep full control of the robot's pitch and yaw, and thus to keep a robust control on the (Ox_Gy_G) plane as shown in figure 11. The thrust is limited by the propellers' intrinsic power, which did not suffice to maintain a constant flight altitude when substantial yaw perturbation occurred.

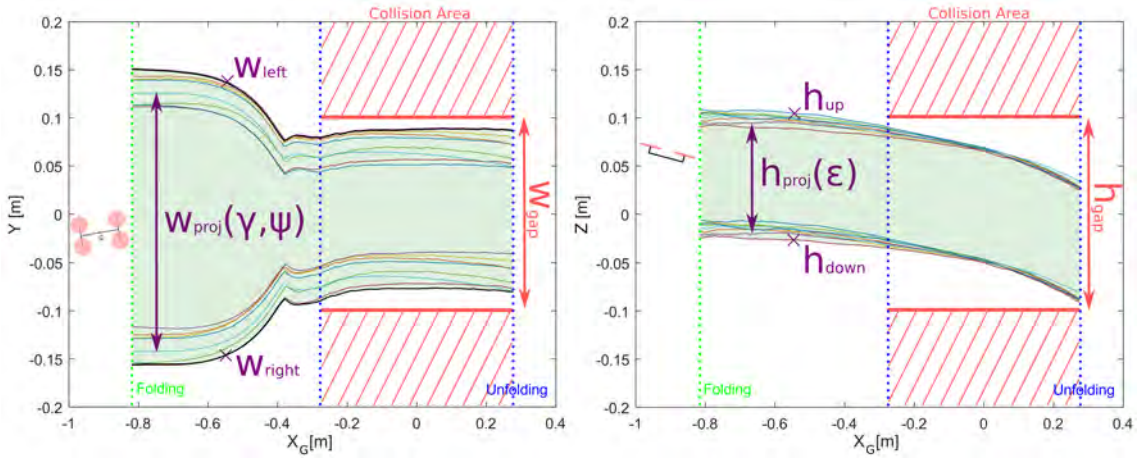


Figure 11: Projected wingspan and height versus relative robot position to the gap

Attitude perturbations

In figure 12 giving Euler angles versus time, we can see that the perturbations encountered are less than 10° on the controllable axis, namely the pitch and yaw axes. These perturbations were not completely rejected because of the lack of thrust power due to the inertia and the weight of the robot in the case of the present platform. It is worth noting that the folding process also induces roll perturbations of about 15°. Attitude perturbations can be explained by inertial effects due to the two arms' acceleration, which did not entirely cancel each other out during the folding process because the arm rotations were not strictly synchronized.

Figure 13 gives data on the angular rates recorded when the robot's position was 2m ahead of the gap and when the robot started folding its wings. As expected in terms of the angular rates, this figure shows that low values (never exceeding $100°/s$) occurred on all the axes before crossing the gap plane.

Comparative results

In order to make quantitative comparisons, we recorded nine unfolded robot flights without the gap at the same speed ($V_X = 2.5m.s^{-1}$) and recorded the results presented in figure 14 and figure 15. This figures give an idea of the robot's performances on lateral axis Y in comparison with the folded robot scenario: it can be seen here that the robot can pass safely through a gap which has a minimum width of $(w_{gap})_{min} = 332mm$. This result can be compared with the minimum gap width $(w_{gap})_{min} = 180mm$ with the folded robot. Minimum passable gaps are plotted in black in figure 15.

The maximum projected wingspan $(w_{proj})_{max}$ observed during all these experiments was also compared with the robot's wingspan while crossing the gap $(\frac{w_{proj}}{l_{ws}})_{max}$. The projected wingspan value (105% of the actual wingspan value) in the morphing robot can be explained by the yaw perturbations which occurred during the folding process.

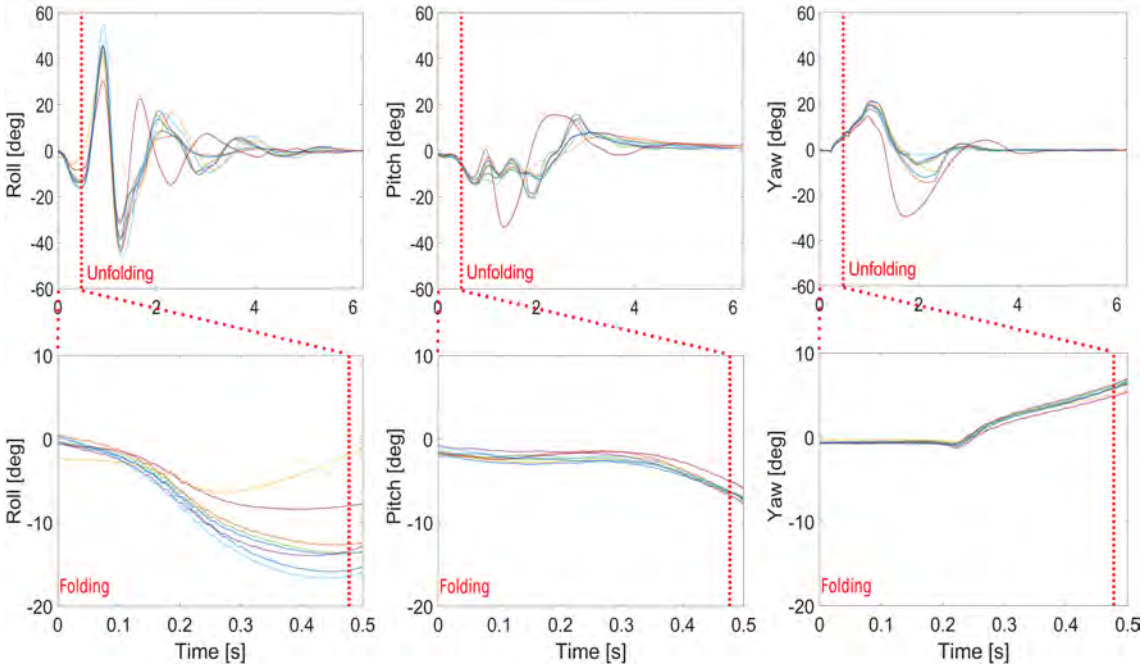


Figure 12: Euler angles during gap crossing and recovery procedure. Time origin corresponds to the beginning of the folding process.

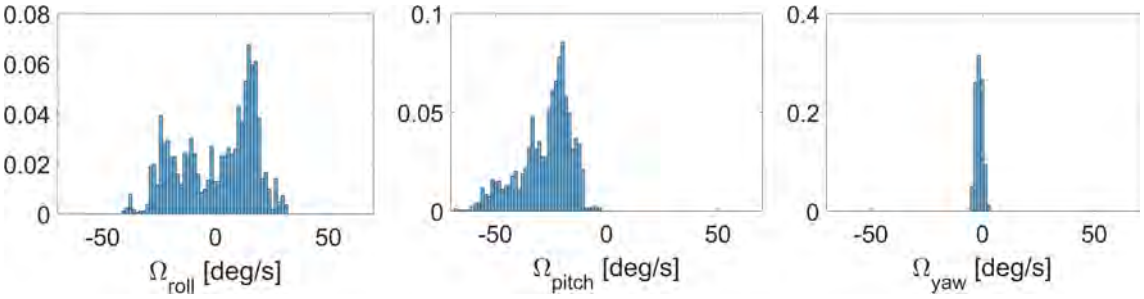


Figure 13: Normalized angular rate distribution before folding in the case of 8 trajectories.

We therefore sought to determine the minimum passable gap width in the case of each configuration (with/without morphing abilities). To quantify the advantages of this novel configuration, we took the ratio between the gap size which could be crossed by the robot with and without morphing abilities:

$$\eta_{meas} = \frac{(w_{gap})_{min,\text{with morph}}}{(w_{gap})_{min,\text{without morph}}} = 54\% \tag{32}$$

This value can be compared with $\eta = 48\%$, which is the theoretical gain in the passable gap width in the case of the morphing robot (as detailed in 2.3).

4 Conclusion

In this paper, a new aerial robotic platform endowed with morphing capabilities is presented. The control laws developed for our Quad-Morphing robot were based on a large body of literature on the non-linear control of quadrotors, including aspects such as attitude estimation and trajectory planning. As observed in studies on birds, our robot can suddenly reduce its wingspan by 50% within a very short time (about 250ms) in order to pass through a gap such as a small square aperture that can be equal to 54% of the robot's unfolded wingspan. We have also presented a gap crossing scenario with a crossing speed as fast as $2.5m.s^{-1}$, which is similar to the forward speed of birds in flight [26].
As bird's wings tuck back against their body, the particularity of our morphing robot is its ability

	$(w_{proj})_{max}$	$(\frac{w_{proj}}{l_{ws}})_{max}$	$(w_{gap})_{min}$
with morphing abilities	$134mm$	105%	$180mm$
without morphing abilities	$268mm$	100%	$332mm$

Figure 14: Results obtained with and without morphing abilities

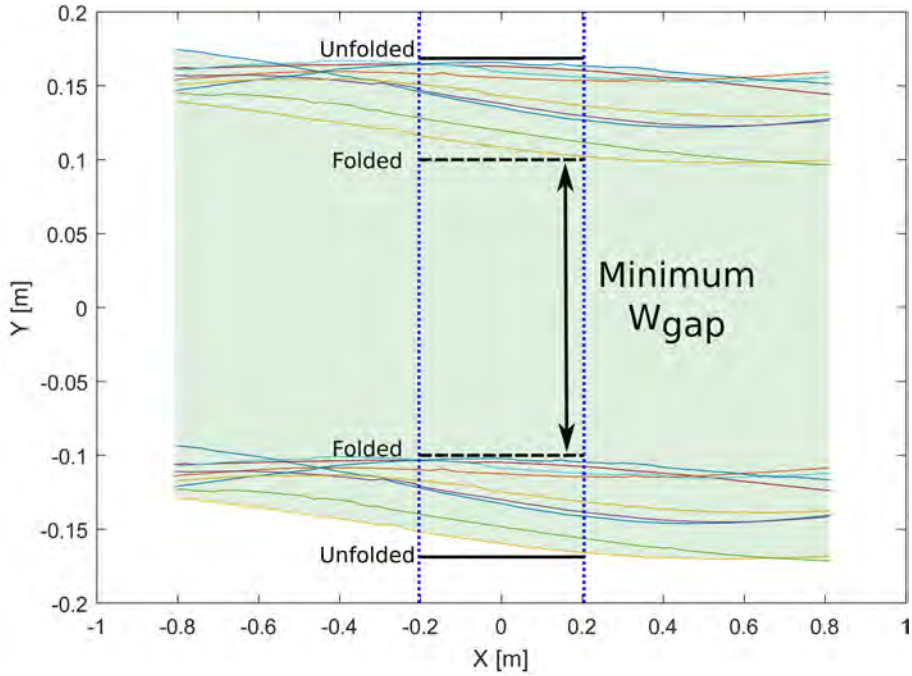


Figure 15: Projected wingspan with unfolded robot versus the robot's position relative to the gap. Two virtual gaps are shown with both configurations: unfolded and folded

to reduce its wingspan, resulting in an inevitable loss of roll control due to the alignment of the four rotors when the robot is in its folded state. During aperture crossing, maximum roll perturbations of up to 15° can occur, but these could be reduced in the future if we can find a means of preserving the robot's roll control when it is in the folded configuration, by tilting the rotors, for example, as proposed by [8].

By adopting new metrics for characterizing the performances of the morphing robot, we have shown that adjusting the yaw angle ($< 7°$ here) directly affects the robot's gap crossing performances (by dramatically increasing the projected wingspan). Better performances could no doubt be achieved by improving the thrust-to-weight ratio (1.5 in the case of our platform) and the rotor's drag coefficient, or by introducing tilting rotors giving yaw-control with thrust.

This novel morphing structure does not have to make aggressive maneuvers to make the aerial robot cross a gap at high speed. Morphing improves the trade-off between payload and fast dynamics. The same morphing principle could be applied to a bigger and heavier quadrotor, assuming that the rotor folding mechanism shows sufficiently fast dynamics. Without having to reduce the payload, it should be possible to equip a future morphing quadrotor with a relatively bulky fixed camera so that autonomous gap crossing can be based entirely on an embedded visual system. Unlike strategies requiring aggressive maneuvers, the angular rates of the Quad-Morphing robot's body consistently reached low values here (always less than $100°/s$) during the approach

phase towards the gap, which can be easily compensated for when implementing an accurate vision-based positioning system. When a robot is flying towards a gap, taking the decision to fold up its structure or not on the basis of visual cues alone will be the next challenge to be met by the Quad-Morphing robot.

Acknowledgement

Authors would like to thank J. Diperi and M. Boyron for their technical support, and J. Blanc for improving the English manuscript. This work was supported by CNRS, Aix-Marseille University, Provence-Alpes-Cote d'Azur region and the French National Research Agency (ANR) with the Equipex/Robotex project.

References

[1] C. Packham, "Goshawk Flies Through Tiny Spaces in Slo-Mo!," *The Animal's Guide to Britain, Episode 3 - BBC Two - YouTube*, 2011.

[2] I. Schiffner, H. D. Vo, P. S. Bhagavatula, and M. V. Srinivasan, "Minding the gap: in-flight body awareness in birds," *Frontiers in Zoology*, vol. 11, no. 1, pp. 64–70, 2014.

[3] M. D. Luca, S. Mintchev, G. Heitz, F. Noca, and D. Floreano, "Bioinspired morphing wings for extended flight envelope and roll control of small drones," *Interface Focus*, vol. 7, no. 1, p. 20160092, 2017.

[4] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robotics and Automation Magazine*, vol. 19, no. 3, pp. 46–56, 2012.

[5] F. Nex and F. Remondino, "UAV for 3D mapping applications: A review," *Applied Geomatics*, vol. 6, no. 1, pp. 1–15, 2014.

[6] D. Shim, H. Chung, H. J. Kim, and S. Sastry, "Autonomous Exploration In Unknown Urban Environments For Unmanned Aerial Vehicles," *Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005.

[7] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[8] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision," *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5774–5781, 2017.

[9] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, Control, and Planning for Aggressive Flight With a Small Quadrotor With a Single Camera and IMU," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 404–411, 2017.

[10] S. Mintchev, S. de Rivaz, and D. Floreano, "Insect-Inspired Mechanical Resilience for Multicopters," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1248–1255, 2017.

[11] S. Mintchev, L. Daler, G. L. Eplattenier, D. Floreano, and S. Member, "Foldable and Self - Deployable Pocket Sized Quadrotor," *Proc. of the IEEE Conference on Robotics and Automation*, pp. 2190–2195, 2015.

[12] A. Desbiez, F. Expert, M. Boyron, J. Diperi, S. Viollet, and F. Ruffier, "X-Morf : a crash-separable quadrotor that morfs its X-geometry in flight," *Proc. of the IEEE International Workshop on Research, Education and Development on Unmanned Aerial Systems (RED-UAS)*, 2017.

[13] N. Zhao, Y. Luo, H. Deng, and Y. Shen, "The Deformable Quad-rotor : Design, Kinematics and Dynamics Characterization, and Flight Performance Validation," *Proc. of the International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[14] G. Barbaraci, "Modeling and control of a quadrotor with variable geometry arms," *Journal of Unmanned Vehicle Systems*, vol. 3, no. 2, pp. 35–57, 2015.

[15] C. Hintz, C. Torno, and L. R. Garcia Carrillo, "Design and dynamic modeling of a rotary wing aircraft with morphing capabilities," *Proc. of the International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 492–498, 2014.

[16] A. Manecy, N. Marchand, F. Ruffier, and S. Viollet, "X4-MaG: A Low-Cost Open-Source Micro-Quadrotor and its Linux-Based Controller," *International Journal of Micro Air Vehicles*, vol. 7, no. 2, pp. 89–109, 2015.

[17] T. Hamel, R. Mahony, R. Lozano, and J. Ostrowski, "Dynamic modelling and configuration stabilization for an X4-flyer," *Proc. of the International Federation of Automatic Control (IFAC)*, vol. 35, pp. 217–222, 2002.

[18] R. Von Mises, "Theory of flight," *Courier Corporation*, pp. 285–290, 1959.

[19] A. Manecy, N. Marchand, and S. Viollet, "RT-MaG: An open-source SIMULINK toolbox for Linux-based real-time robotic applications," *Proc. of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 173–180, 2014.

[20] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear Complementary Filters on the Special Orthogonal Group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[21] D. Brescianini, M. Hehn, and R. D'Andrea, "Nonlinear Quadrocopter Attitude Control," *Department of Mechanical and Process Engineering, ETHZ, Tech. Rep.*, 2013.

[22] G. Christopher, G. Ricardo, and R. Andrew, "Quaternion-Based Hybrid Control for Robust Global Attitude Tracking," *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2555–2566, 2011.

[23] D. Mellinger and V. Kumar, "Trajectory Generation and Control for Quadrotors," *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2520–2525, 2011.

[24] K. P. Tee and S. S. Ge, "Control of nonlinear systems with full state constraint using a barrier lyapunov function," *Proc. of the IEEE Conference on Decision and Control*, pp. 8618–8623, 2009.

[25] M. Faessler, F. Fontana, C. Forster, and D. Scaramuzza, "Automatic re-initialization and failure recovery for aggressive flight with a monocular vision-based quadrotor," *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1722–1729, 2015.

[26] H. D. Vo, I. Schiffner, and M. V. Srinivasan, "Anticipatory Manoeuvres in Bird Flight.," *Scientific reports*, vol. 6, p. 27591, 2016.