Extending Guiding Vector Field to track unbounded UAV paths

Mael Feurgard, Gautier Hattenberger, Simon Lacroix

Extending Guiding Vector Field to track unbounded UAV paths

Mael Feurgard[1], Gautier Hattenberger[2], Simon Lacroix[1]

***Abstract*— A recent advance in vector field path following is the introduction of the *Parametric Guiding Vector Field* method. It allows for singularity-free vector fields with strong convergence guarantees, usable even for self-intersecting paths. However, the method requires significant gain tuning for practical use. In particular, for unbounded paths, the gains will inevitably become ill-suited for efficient path following. We propose a method to overcome this issue by introducing a dynamic step adaptation strategy, which provides additional normalization properties to the field. This allows the following of unbounded curves and reduces the number of gains to tune. The proposed improvements are verified in simulations using the PaparazziUAV software.**

I. Introduction

Path following is a fundamental problem in autonomous robotics, and numerous methods have been developped to solve it [1, 2]. Among them, *vector field path following* has seen interesting recent development [3–6].

In [4], the focus is on designing a vector field path following algorithm for UAVs robust to perturbations such as wind, but the paths are limited to lines and circles. In [3] and [5], path following of arbitrary curves is proposed, respectively for unicycle robots in 2D and fixed-wing UAVs in 3D. But both require specific treatments as the methods create fields with singularities. [3] handles this issue by ensuring that the singularities are far from the path, and the robot, if starting close enough, will stay near the path. In [5], singularities are handled by maintaining the controller to the last non-singular command when in the vicinity of a singularity.

More importantly, these designs cannot handle self-intersecting curves. This specific problem, with the more general issue of singular points in the field, have been solved by the *Parametric Guiding Vector Field* algorithm [6]. It is a path-following strategy that generates a dynamic vector field based on a virtual tracking point. It has been designed first with fixed-wing UAVs in mind [6], then successfully tested in real conditions [7]. It has also been applied to unmanned surface vessels [8], and extended to include coordinated path following [8, 9].

Nevertheless, this algorithm requires a number of gains to be tuned for practical use. Furthermore, these gains make the algorithm unable to deal with curves with unbounded derivatives. In this paper, we introduce a modification to this algorithm, called *step adaptation*, to better handle these unbounded curves and simplify gain tuning.

[1] Laboratoire d'Analyse et d'Architecture des Systèmes, Université de Toulouse, CNRS, Toulouse, France

[2] Fédération ENAC ISAE-SUPAERO ONERA, Université de Toulouse, France

Section II introduces the *Parametric Guiding Vector Field* algorithm, its benefits and limitations. Section III motivates and presents the proposed improvements to the algorithm. Section IV finally validates the improvements by running simulations using PaparazziUAV [10].

II. Presentation of the Parametric Guiding Vector Field algorithm

We first summarize the general concepts of the *Parametric Guiding Vector Field* algorithm, introduced in [6]. Then we delve into the details of its practical implementation to discuss its advantages and inconvenients.

A. General description

The *Parametric Guiding Vector Field* algorithm (abbr. *P-GVF*) is a path following algorithm that tracks a point on the curve to guide the robot and that controls this tracking point along the curve depending on the robot location. By combining these two elements, it achieves path following, even for self-intersecting curves.

P-GVF can be applied to any n-dimension path described by a twice continuously differentiable function $f \in \mathcal{C}^2(\mathbb{R}, \mathbb{R}^n)$. It defines a *guiding point* $f(w) \in \mathbb{R}^n$ identified by its *virtual coordinate* $w \in \mathbb{R}$ for the robot located at $p \in \mathbb{R}^n$ to follow. Let $\xi = (w, p) \in \mathbb{R} \times \mathbb{R}^n$ the *extended coordinate*, identitying simultaneously the *guiding point* and robot locations p. *P-GVF* defines a vector field $\chi \colon \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ to drive ξ with respect to time t, that is such that

$$\frac{\mathrm{d}\xi}{\mathrm{d}t} = \chi(\xi(t)) \tag{1}$$

χ updates both the *guiding point* and robot locations simultaneously, aiming to have the robot track the *guiding point* and the *guiding point* move along the path in a way ensuring both tracking and progress along the path.

χ is mostly defined through the *error function* $\phi \colon \mathbb{R}^{n+1} \to \mathbb{R}^n$, itself defined as $\phi(\xi) = p - f(w)$. For p,f and ϕ, we denote their respective individual components $p = (p_1, \dots, p_n)$, $f = (f_1, \dots, f_n)$ and $\phi = (\phi_1 \dots, \phi_n)$, such that $\phi_i(\xi) = p_i - f_i(w)$. χ is defined as

$$\chi = \times(\nabla\phi_1, \dots, \nabla\phi_n) - \sum_{i=1}^{n} k_i \phi_i \nabla\phi_i \tag{2}$$

with $\nabla\phi_i$ the gradient of ϕ_i with respect to ξ, $\times(\cdot)$ the generalized cross product as defined in [11, Chapter 7.2], and $k_1, \dots, k_n$ some positive gains.

The second component of the field $(-\sum_{i=1}^{n} k_i \phi_i \nabla\phi_i)$ is a weighted attractor to the *guiding point* $f(w)$, whereas the first is orthogonal to all $\nabla\phi_i$ ([11, Proposition 7.2.1]), and

guides the whole system along the path. The gains $k_1, \dots, k_n$ can be used to balance between attraction to the guiding point and following the path orientation, and this for each dimension.

The main advantage of this field is that it ensures theoretical asymptotic convergence to the guiding point regardless of the initial position for all $\mathcal{C}^2$ paths, in particular self-intersecting ones. But as defined here, it lacks several elements to be used practically. Indeed, as a field based approach, it requires the addition of a controller for all nonholonomic robots. Furthermore, the exact definition of the parametric curve describing the path has a significant impact on the field which makes it unpractical. This is balanced by introducing additional gains.

B. Practical considerations

We develop the definitions introduced previously to give a better insight in the different aspects of the guiding field. We then introduce the gains used to make the field practical, and discuss their respective roles.

We can rewrite the main equation (1) to split it into the *virtual field*, noted χ^{virt} and the *physical field*, χ^{phys}, respectively in charge of updating the virtual coordinate and robot position:

$$\begin{cases} \dfrac{\mathrm{d}w}{\mathrm{d}t}(t) = \chi^{virt}(w,p)(t) \\ \dfrac{\mathrm{d}p}{\mathrm{d}t}(t) = \chi^{phys}(w,p)(t) \end{cases} \tag{3}$$

With $K = \mathrm{diag}(k_1, \dots, k_n) \in \mathcal{M}_{n\times n}(\mathbb{R})$, we can simplify their respective expressions to obtain:

$$\chi^{virt}(w,p) = (-1)^n + f'(w)^T K \phi(w,p) \tag{4}$$
$$\chi^{phys}(w,p) = (-1)^n f'(w) - K\phi(w,p) \tag{5}$$

By definition of ϕ, the *physical field* χ^{phys} is an attractor to the *guiding point*, deformed by K and translated by $(-1)^n f'(w)$. Still, χ^{phys} has a centerpoint where is cancels, we call it the *attractor point* and denote it p_0. We have $p_0 = f(w) + (-1)^n K^{-1} f'(w)$.

To better illustrate the *virtual field*, we introduce the *displacement field* χ^{displ} defined as

$$\chi^{displ}(w,p) = f(w + \chi^{virt}(w,p)) - f(w)$$

It is useful as it can be directly plotted to describe the dynamics of the *guiding point* depending on the robot location (see Figure 1).

There are two main elements in the expressions of χ^{virt} (4) and χ^{phys} (5): the error function $\phi(w,p)$ and the tangent to the curve at the *guiding point* $f'(w)$. Using f' is useful as it gives a direction along the curve, but it is also problematic as its magnitude can vary widely depending on the parametrization used for the curve. More generally, the dynamic of the *guiding point* is highly dependant on the curve parametrization.

Hence, two additional gains, $\beta \in \mathbb{R} \setminus \{0\}$ and $L \in \mathbb{R}_{>0}$ are introduced:

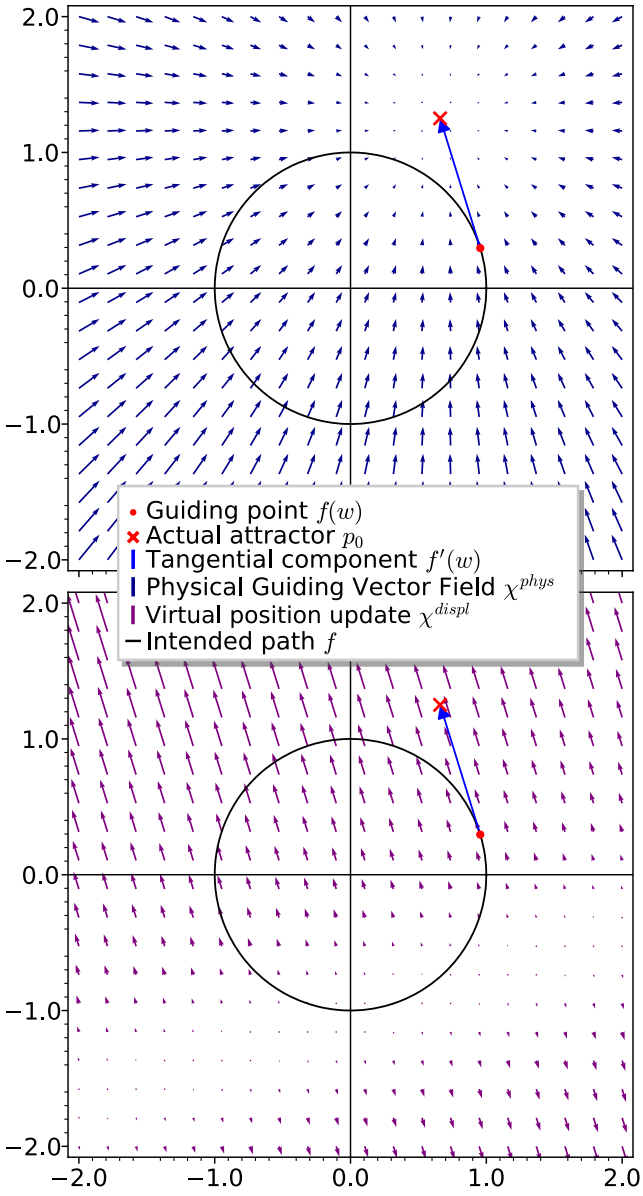

Fig. 1: Illustration of the entire Parametric GVF through its physical field χ^{phys} (top) and its displacement field χ^{displ} (bottom) for a circle $f(w) = (\cos(w), \sin(w))$. Here $w = 0.3$ and $K = I_2$

- $\beta \in \mathbb{R} \setminus \{0\}$, parametric rescale:

$$f(w) \mapsto f(\beta \cdot w)$$

It is intended as a simple way to re-parametrize the curve and tune its derivative. It affects the fields as follow:

$$\chi^{phys}(w,p) = (-1)^n \cdot \beta \cdot f'(\beta w) - K\phi(\beta w, p)$$
$$\chi^{virt}(w,p) = (-1)^n + \beta \cdot f'(\beta w)^T K \phi(\beta w, p)$$

It modifies the balance between the tangential and attractive components in the physical field. But most importantly, it modifies the dynamics of the virtual component, allowing to scale up or down the scale of

the *displacement field*. Having $|\beta| > 1$ has a tendency to increase the speed of the guiding point, whereas $|\beta| < 1$ reduces it. Taking β negative allows to change the orientation of the tangent. This is especially useful to match this orientation with the one defined by the constant component $(-1)^n$ of χ^{virt}.

- $L \in \mathbb{R}_{>0}$, error function rescale:

$$\phi(w,p) \mapsto L \cdot \phi(w,p)$$

It rescales the whole field, but non-linearly:

$$\chi^{phys}(w,p) = (-L)^n \cdot \beta \cdot f'(\beta w) - K \cdot L \cdot \phi(\beta w, p)$$
$$\chi^{virt}(w,p) = (-L)^n + L \cdot \beta \cdot f'(\beta w)^T K \phi(\beta w, p)$$

With the other gains added, the principal change provided by L is the ability to tune the bias used in χ^{virt}. This is particularly helpful combined with β to ensure a good equilibrium between the variable intensity of the steps, guided by $f'(\beta w)$, and its constant intensity, $(-L)^n$.

One can argue that the gains L and β yield unwieldy expressions for the fields χ^{virt} and χ^{phys}. They are defined by altering f and ϕ instead of the field construction (2). Hence, the field can be tuned without altering the convergence result.

To summarize, the whole field with the gains present is:

$$\chi^{phys}(w,p) = (-L)^n \cdot \beta \cdot f'(\beta w) - L \cdot K \cdot \phi(\beta w, p) \quad (6)$$
$$\chi^{virt}(w,p) = (-L)^n + L \cdot \beta \cdot f'(\beta w)^T K \phi(\beta w, p) \quad (7)$$

Note that these gains change the formula for the physical position p_0 of the attractor making up χ^{phys}:

$$p_0 = f(\beta w) + (-L)^n \cdot \beta \cdot K^{-1} \cdot f'(\beta w) \quad (8)$$

This abundance of gains can make *Parametric GVF* difficult to use as tuning should be made with regard to the parametric curve and robot simultaneously.

III. Normalizing curves for GVF

The main issue of *P-GVF* is that for some curves, f' may be unbounded. Hence, the constant gain β will inevitably become ill-suited during the execution. To solve this problem, we propose to dynamically approximate a reparametrization of f guaranting $\|f'\|$ to be constant.

First, we present how using a normalized parametric curve simplifies the *Parametric GVF* algorithm. Then, we show that for any regular parametric curve, it is possible to compute a normalized equivalent. And finally, we implement in *Parametric GVF* our changes, which allow considering a normalized equivalent of the curve given any input.

To simplify the description, we limit ourselves to the 3D case ($n = 3$). Through the following sections, we consider a parametric curve $f \in \mathcal{C}^2(\mathbb{R}, \mathbb{R}^3)$.

A. Parametric GVF with a normalized curve

A parametric curve $g\colon \mathbb{R} \to \mathbb{R}^3$ is said to be a *normalized curve* if, for all $s \in \mathbb{R}$

$$\left\| \frac{\mathrm{d}g}{\mathrm{d}s}(s) \right\| = 1 \quad (9)$$

For such a curve, the parameter s in called *curvilinear abscissa*, as a unit variation in parameter $\mathrm{d}s$ is equal in length to a unit variation in space $\mathrm{d}g$.

Let us now consider *Parametric GVF* but with the additional hypothesis that the path is described by a normalized curve g. We rewrite the main fields (6) and (7):

$$\chi^{phys}(w,p) = -L^3 \beta \cdot g'(\beta w) - LK \cdot \phi(\beta w, p) \quad (10)$$
$$\chi^{virt}(w,p) = -L^3 + L \left\| \beta \cdot K \cdot \phi(\beta w, p) \right\| \cos(\alpha) \quad (11)$$

The change is immediately significant for χ^{virt}, as its variations depend only on two geometric values: the distance to the guiding point $\|\phi(\beta w, p)\|$ and the angle α between the tangent $g'(\beta w)$ and the error vector $\phi(\beta w, p)$.

For χ^{phys}, one observable consequence is the distance between the attractor point p_0 (defined in (8)) and the guiding point $g(\beta w)$:

$$\|p_0 - g(\beta w)\| = L^2 \left|\beta\right| \cdot \left\| K^{-1} g'(\beta w) \right\|$$

For a scalar gain matrix $K = k \cdot I_3, \quad k \in \mathbb{R}_{>0}$, the expression simplifies and becomes independent from w:

$$\|p_0 - g(\beta w)\| = L^2 \left| \frac{\beta}{k} \right|$$

When K is not a scalar matrix, we get $\|p_0 - g(\beta w)\| \leq L^2 \left|\beta\right| \left\| K^{-1} \right\|_2$, with $\|\cdot\|_2$ the induced matrix norm from the Euclidean norm.

Hence, we argue that using a *normalized curve* simplifies the fields used in *Parametric GVF*, as their definitions can be described mostly in terms of physical, geometric values.

B. Normal parametrization of a curve

In this section, we show that for any parametric curve f which is $\mathcal{C}^2$ and never stationary (i.e. there are no interval $I \subset \mathbb{R}$ such that $f'(I) = \{\mathbf{0}\}$), there exists a bijection σ such that $g = f \circ \sigma$ is *normalized*.

We define the *arc-length* function γ_f for all $w \in \mathbb{R}$ as

$$\gamma_f(w) = \int_0^w \|f'(t)\| \ dt \quad (12)$$

γ_f is a bijection, since its derivative $w \mapsto \|f'(w)\|$ is nonnegative and is null on no interval. Hence, the reciprocal γ_f^{-1} exists, and we can define $g = f \circ \gamma_f^{-1}$. Using the chain rule and the inverse function derivation rule, we get:

$$g' = \frac{f' \circ \gamma_f^{-1}}{\gamma_f' \circ \gamma_f^{-1}}$$

$\gamma_f'(w) = \|f'(w)\|$, so we can conclude that $\|g'(w)\| = 1$ for all $w \in \mathbb{R}$; g is a normal parametrization of f.

Unfortunately, obtaining a closed analytical expression for γ_f then γ_f^{-1} is very often impossible to do. Nevertheless, we

can numerically compute an approximation of γ_f^{-1}, which we shall call σ from now on.

C. Approximating a normal parametrization

Let $w(t)\colon \mathbb{R}_{\geq 0} \to \mathbb{R}$ the temporal evolution of the *virtual coordinate*, driving the evolution of the *guiding point* $f(w(t))$. This can be interpreted as a re-parametrization over time, yielding $g = f \circ w$. The speed at which the guiding point moves with respect to time is given by

$$\begin{aligned}\left\|\frac{\mathrm{d}g}{\mathrm{d}t}(t)\right\| &= \left\|\frac{\mathrm{d}(f\circ w)}{\mathrm{d}t}(t)\right\| \\ &= \left|\frac{\mathrm{d}w}{\mathrm{d}t}(t)\right| \cdot \left\|\frac{\mathrm{d}f}{\mathrm{d}w}(w(t))\right\|\end{aligned}$$

Hence, note that if f is already normally parametrized, the actual speed of the *guiding point* $\left\|\frac{\mathrm{d}g}{\mathrm{d}t}(t)\right\|$ is directly $\left|\frac{\mathrm{d}w}{\mathrm{d}t}\right|$. By setting $\left|\frac{\mathrm{d}w}{\mathrm{d}t}\right|$ to the aircraft speed, we can ensure that the *guiding point* evolution matches the aircraft one.

Now, suppose f is not normally parametrized. We desire that the *guiding point* evolution matches the aircraft one, and have to adapt the evolution of w. Using finite elements approximation, we can reformulate the previous equation:

$$\begin{aligned}&\left\|\frac{g(t+\Delta t)-g(t)}{\Delta t}\right\| = \left|\frac{\Delta w}{\Delta t}\right| \cdot \left\|\frac{f(w+\Delta w)-f(w)}{\Delta w}\right\| \\ \Leftrightarrow\quad &\|g(t+\Delta t)-g(t)\| = \|f(w+\Delta w)-f(w)\|\end{aligned}$$

By defining $|\Delta s| := \|g(t+\Delta t)-g(t)\|$ the desired spatial step at time t, we finally obtain:

$$|\Delta s| = \|f(w+\Delta w)-f(w)\| \tag{13}$$

Based on this approximation, given at each time t a desired physical step Δs, we have to compute the corresponding Δw respecting (13) ensuring that the *guiding point* dynamics match the expected one, regardless of the parametrization of f. Note that we used $|\Delta s|$ instead of Δs directly in order to encode an orientation. Indeed, this equation specifies a step *length*, but no *direction*, i.e. a sign for Δw. Hence, to achieve any parametrization specified through a sequence a physical step Δs, we can solve the following system at each time step:

$$\begin{cases} |\Delta s| &= \|f(w+\Delta w)-f(w)\| \\ 0 &< (\Delta s)(\Delta w) \end{cases} \tag{14}$$

Informally, this amount to, given some position on the curve and a distance to travel, find the next position on the curve such that its distance from the current one is equal to the one specified, and in the desired direction.

Since f is twice continuously differentiable on $\mathbb{R}$, we can approximate $f(w+\Delta w)-f(w)$ using a second order Taylor development, yielding:

$$|\Delta s| \simeq \left\|(\Delta w)f'(w) + \frac{f''(w)}{2}(\Delta w)^2\right\| \tag{15}$$

Note that with a first order development, obtaining the matching Δw is immediate, as long as the curve is not singular, i.e. $f'(w) \neq 0$.

By squaring, we can develop the norm in (15), resulting in a polynomial equation in Δw. Assuming f is never 2-singular, that is there are no $w \in \mathbb{R}$ such that both $f'(w) = 0$ and $f''(w) = 0$, this equation always has at least one positive and one negative solution. We focus on finding the smallest in absolute value satisfying $0 < (\Delta s)(\Delta w)$, hence defining a function giving approximate solutions of (14). We call this function *step_adaptation*. It can be computed efficiently using a polynomial root finder algorithm.

D. Integration into Parametric GVF

As shown in III-A, it is preferable to use a normalized curve than an arbitrary one in the *Parametric GVF* algorithm. Given f, we denote g a normalized equivalent, defined through an unknown reparametrization function σ such that $g = f \circ \sigma$. We show in this section how it is nevertheless possible to consider g instead of f in the algorithm by using our *step_adaptation* function.

We denote $u \in \mathbb{R}$ the parameter provided to g, which is not directly known, and $w = \sigma(u) \in \mathbb{R}$ the parameter given to f, which is known. The first thing we need is to be able to compute the derivatives of g (g'' may be required by the underlying controller). Since $\|g'(u)\| = 1$ for all $u \in \mathbb{R}$, we have the identity:

$$\sigma'(u) = \frac{1}{\|f'(\sigma(u))\|} \tag{16}$$

(or equivalently, $\sigma'(u) = -1/\|f'(\sigma(u))\|$. Note that the sign cannot alternate arbitrarily, as it would contradict continuity). We can exploit it to compute $g'(u)$ and $g''(u)$ as functions of $f(\sigma(u)), f'(\sigma(u)), f''(\sigma(u))$.

We still need to update the virtual parameter we have access to, that is $w := \sigma(u)$, without an explicit expression for σ. We can achieve this by using *step adaptation*.
We take Δs as defined by the virtual field based on g, and adapt it to the aircraft speed by rescaling the step using its ground speed v and the time step Δt at which the algorithm is called. We use this value to drive the *step adaptation* for f, computing a Δw ensuring (14).

Hence, we obtain a dynamic for the *guiding point* driven by *Parametric GVF*, following the normally parametrized curve g, having the same shape as f. The method is summarized by the pseudocode algorithm (algorithm 1), with the differences introduced by normalization in red.

We have depicted a way to transform any parametric curve f into an equivalent normalized one in the context of the *Parametric GVF* algorithm. It ensures that the resulting curve has a bounded derivative, with constant norm. We call this new algorithm *Normalized GVF*.

Algorithm 1: N-GVF main loop

input : Values for parameters L, β, K; initial w and function $f : \mathbb{R} \to \mathbb{R}^n$
output: Commands to controller

begin
  Initialize w
  $t \leftarrow get\_time()$
  repeat
    $p, v \leftarrow get\_state()$
    $\phi \leftarrow p - f(\beta w)$
    $\chi^{phys} \leftarrow$
     $(-L)^n \cdot \beta \cdot f'(\beta w) / \|f'(\beta w)\| - K \cdot L \cdot \phi$
    $\chi^{virt} \leftarrow (-L)^n + L \cdot \beta \cdot f'(\beta w)^T K \phi / \|f'(\beta w)\|$
    $curr\_t \leftarrow get\_time()$
    $\Delta s \leftarrow v \cdot (curr\_t - t) \cdot \chi^{virt}$
    $w \leftarrow w + step\_adaptation(\Delta s, f, \beta w)$
    Provide χ^{phys} to controller
    $t \leftarrow curr\_t$
  until *End Of Time*

IV. Experiments

We argue that the new properties introduced with *Normalized GVF* ease the tuning of the algorithm, make it less sensitive, and allows to tackle more efficiently unbounded paths. To validate our improvements, we compare the former *Parametric GVF* with our new *Normalized GVF* in simulations. We use the 6 Degrees of Freedom simulator from PaparazziUAV [7, 12]. The controller, is the one introduced in [6, Section VI.B].

A. Simulation methodology

We present the results of two experiments using self intersecting curves. For each experiment and each method, we deploy three identical fixed wing aircrafts, starting from the same takeoff point (collisions are disabled), in an open area (no obstacles, flat ground). They are all tuned with slightly different gains in order to study their respective impact and sensitivity. The tuning is based on previous experiments if existing, and otherwise manually empirically determined.

The two different paths used for testing are:

- 3D Lissajous: used both in [6, Section VI.E] and [9, Section VII.B], it serves as a baseline to ensure performances are not diminished and to illustrate a simplification in gain tuning. It is defined as:

$$f(w) = \begin{bmatrix} a_x \sin(\omega_x w + \varphi_x) \\ a_y \sin(\omega_y w + \varphi_y) \\ a_z \sin(\omega_z w + \varphi_z) \end{bmatrix}$$

  with $a_x = a_y = 225\text{m}$, $a_z = -20\text{m}$, $\omega_x = 1$, $\omega_y = \omega_z = 2$, and $\varphi_x = \varphi_z = 0$, $\varphi_y = \pi/2$.
- Drifting Ellipse (Figure 2): it is equivalent to following a point on a XY ellipse which is being scaled up while drifting along the X-axis. It is defined as:

$$f(w) = \begin{bmatrix} a_x \cdot w \cdot \cos(2\pi f w) + v_x \cdot w \\ a_y \cdot w \cdot \sin(2\pi f w) \\ 0 \end{bmatrix}$$

with $v_x = 5\text{m}$, $a_x = 2\text{m}$, $a_y = 4\text{m}$, $f = 0.05$.

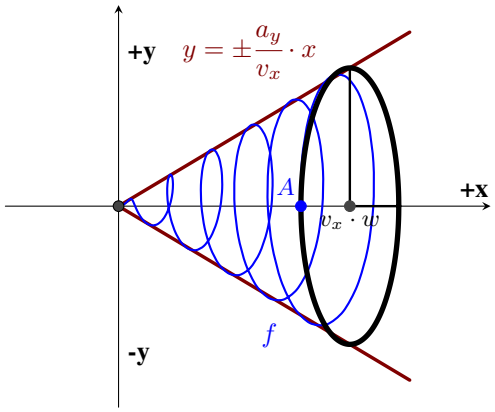

Fig. 2: Illustration of a Drifting Ellipse curve

Our main metric is the Euclidean distance between the aircraft and its associated *guiding point* $f(w)$, given by $\|\phi(w, p)\|$, that measures the accuracy of the path following.

To complement this metric, we define a summarizing value. We consider that the "asymptotic" part of our time series are reached for the second half of the experiment. We use this part to define the asymptotic mean Euclidean distance $\overline{\|\phi\|}$. These values should be taken with caution, as it is difficult to reduce the time series to a single value. Nevertheless, we use them to complement our comparison when needed.

B. Results analysis

The graphs for $\|\phi(w, p)\|$ are given in Figure 3, top graph for 3D Lissajous experiment and bottom for Drifting Ellipse. A summary of the best results with the gain configurations is given in Table I.

For the 3D Lissajous experiment (Figure 3, top), the general behavior is similar across all aircraft, namely a periodicity in the values taken by the euclidean distance. This is due to overshooting by all aircraft at the sharp turns of the curve. Still, note that because of the different gain tunings, not all aircraft behave similarly. For instance, among the three running with *Normalized GVF* (AC 21,22,23), AC 21 has a significantly higher error on average, and AC 23 oscillates more often. These are respectly due to under and overfitting the gains for this path following task. On the other hand, the performances of the aircraft using *Parametric GVF* are quite similar despite the different gain configurations.

For the Drifting Ellipse experiment (Figure 3, bottom), we observe unstable behavior using *Parametric GVF*. We suppose this is due to a specific balance between the gains and curve derivative, leading to a first augmentation followed by a stronger divergence in the error. The latter is most likely due to numeric instability in the update of the *guiding point*

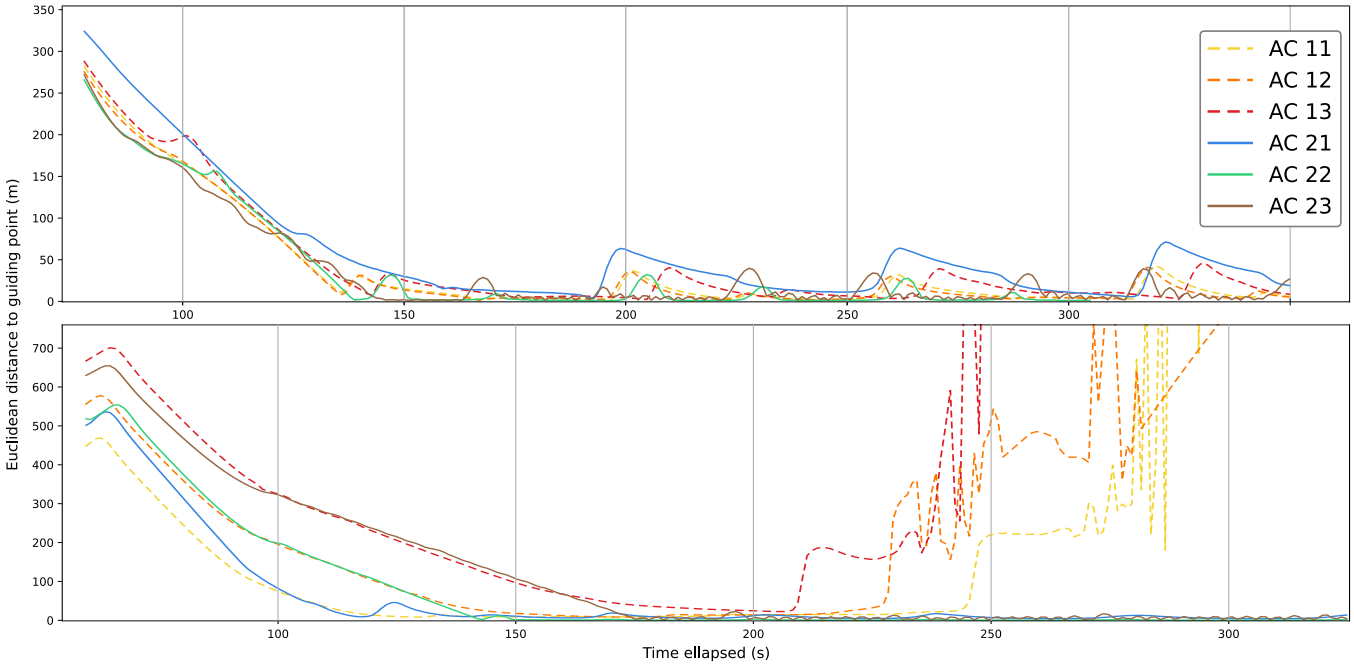

Fig. 3: Euclidean distance to the guiding point for the two experiments (3D Lissajous top, Drifting Ellipse bottom) AC 11,12,23 are running *Parametric GVF*; AC 21,22,23 are running *Normalized GVF*

location. Regarding the aircraft running *Normalized GVF*, there are no major issues. Still, we notice the same flaws as in the 3D Lissajous experiment, namely oscillations for AC 23 and understeering for AC 21.

For each experiment and method, we decide which aicraft performs best by first looking at the time series for the Euclidean distance, then the averaged error $\overline{\|\phi\|}$. In the Drifting Ellipse experiment, for the aircraft running with *Parametric GVF*, we select AC 11 as best, being the last to diverge. The summary of results is given in Table I.

TABLE I: Summary of best results (3D-L for 3D Lissajous, D-E for Drifting Ellipse)

	Method	ID	β, L, k_1, k_2, k_3	$\overline{\|\phi\|}$ (m)
3D-L	*P-GVF*	12	$0.03, 0.55, 0.042, 0.042, 0.0525$	8.4
	N-GVF	22	$\mathbf{1}, \mathbf{1}, 0.042, 0.042, 0.0525$	3.9
D-E	*P-GVF*	11	$0.5, 0.2, 0.01, 0.01, 0.01$	–
	N-GVF	22	$\mathbf{1}, \mathbf{1}, 0.06, 0.06, 0.06$	1.2

The main result is that *Normalized GVF* maintains convergence when *Parametric GVF* does not in the case of the unbounded Drifting Ellipse curve. This illustrates both the inherent limits of *Parametric GVF* and the success of the proposed modifications. It can also be observed that in our experiments, the best aircraft using *Normalized GVF* has two fewer gains and performs always better than its *Parametric GVF* counterpart. There may exist better tunings for both algorithms with respect to the curves used, but it is important to note that there are no known automatic tuning strategy other than empirical testing. Hence we argue that using *Normalized GVF* reduces the number of gains to tune, simplifying tuning for this method.

V. Conclusions and Future Works

Dynamic normalization of a curve has been introduced in the *Parametric Guiding Vector Field* algorithm. This new method, called *Normalized Guiding Vector Field*, allows path following for unbounded self-intersecting paths. Additionally, it reduces the number of gains to tune for using this method in practice.

There are still significant limits to this method. Even if there are less gains to tune, gain tuning is still required and non-trivial. Nevertheless, we believe it is possible to build an automatic tuning method based on a normalized curve, as the gains impact can be geometrically measured. This geometric interpretation can then be linked to the robot characteristics. Another important limit concerns the simulations. Indeed, they were performed without wind, which has a significant impact on small UAVs. Still, we think this is not too much of an issue, as real experiments in windy conditions were done with success using *Parametric GVF* [7], and we did not modify the controller nor the general shape of the field. Furthermore, the method can be applied to other vehicules, such as Unmanned Surface Vessels [8].

In future work, we will focus on adapting *Normalized GVF* to the coordinated path following problem. Indeed, *Parametric GVF* has already been extended to this end [8, 9]. But the coordination is mostly based on the abstract *virtual coordinate*. As it is not linked to any physical value, it makes tuning more difficult. We hope that, using normalized curves, we will be able to implement coordination in *Parametric GVF* related to geometric values.

Acknowledgments: Work partially founded by the ANR Projet “Panache” (ANR-21-ASRO-0007)

References

[1] Bartomeu Rubí, Ramon Pérez, and Bernardo Morcego. "A Survey of Path Following Control Strategies for UAVs Focused on Quadrotors". In: *Journal of Intelligent & Robotic Systems* 98.2 (May 2020), pp. 241–265. ISSN: 1573-0409. DOI: 10.1007/s10846-019-01085-z. URL: https://doi.org/10.1007/s10846-019-01085-z.

[2] Haitong Xu and C. Guedes Soares. "Review of Path-following Control Systems for Maritime Autonomous Surface Ships". In: *Journal of Marine Science and Application* 22.2 (June 2023), pp. 153–171. ISSN: 1993-5048. DOI: 10.1007/s11804-023-00338-6. URL: https://doi.org/10.1007/s11804-023-00338-6.

[3] Yuri A. Kapitanyuk, Anton V. Proskurnikov, and Ming Cao. "A Guiding Vector-Field Algorithm for Path-Following Control of Nonholonomic Mobile Robots". In: *IEEE Transactions on Control Systems Technology* 26.4 (2018), pp. 1372–1385. DOI: 10.1109/TCST.2017.2705059.

[4] Ximan Wang et al. "Adaptive Vector Field Guidance Without a Priori Knowledge of Course Dynamics and Wind". In: *IEEE/ASME Transactions on Mechatronics* 27.6 (2022), pp. 4597–4607. DOI: 10.1109/TMECH.2022.3160480.

[5] Adriano M.C. Rezende et al. "Robust Fixed-Wing UAV Guidance with Circulating Artificial Vector Fields". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 5892–5899. DOI: 10.1109/IROS.2018.8594371.

[6] Weijia Yao et al. "Singularity-Free Guiding Vector Field for Robot Navigation". In: *IEEE Transactions on Robotics* 37.4 (2021), pp. 1206–1221. DOI: 10.1109/TRO.2020.3043690.

[7] Hector Garcia de Marina, Murat Bronz, and Gautier Hattenberger. *Guiding vector fields in Paparazzi autopilot*. 2021. DOI: 10.48550/ARXIV.2106.10680.

[8] Bin-Bin Hu et al. "Spontaneous-Ordering Platoon Control for Multirobot Path Navigation Using Guiding Vector Fields". In: *IEEE Transactions on Robotics* 39.4 (2023), pp. 2654–2668. DOI: 10.1109/TRO.2023.3266994.

[9] Weijia Yao et al. "Guiding Vector Fields for the Distributed Motion Coordination of Mobile Robots". In: *IEEE Transactions on Robotics* 39.2 (Apr. 2023), pp. 1119–1135. ISSN: 1941-0468. DOI: 10.1109/tro.2022.3224257.

[10] Balazs Gati. "Open source autopilot for academic research - The Paparazzi system". In: *2013 American Control Conference*. 2013, pp. 1478–1481. DOI: 10.1109/ACC.2013.6580045.

[11] Antonio Galbis and Manuel Maestre. "Vectors and Vector Fields". In: *Vector Analysis Versus Vector Calculus*. Boston, MA: Springer US, 2012, pp. 1–17. ISBN: 978-1-4614-2200-6. DOI: 10.1007/978-1-4614-2200-6_1. URL: https://doi.org/10.1007/978-1-4614-2200-6_1.

[12] Gautier Hattenberger, Murat Bronz, and Michel Gorraz. "Using the Paparazzi UAV System for Scientific Research". In: *IMAV 2014, International Micro Air Vehicle Conference and Competition 2014*. Delft, Netherlands, Aug. 2014, pp 247–252. DOI: 10.4233/uuid:b38fbdb7-e6bd-440d-93be-f7dd1457be60. URL: https://enac.hal.science/hal-01059642.