Guidance, navigation and control algorithms for autonomous agricultural systems

M. Mammarella[1], C. Donati[2], F. Dabbene[1]

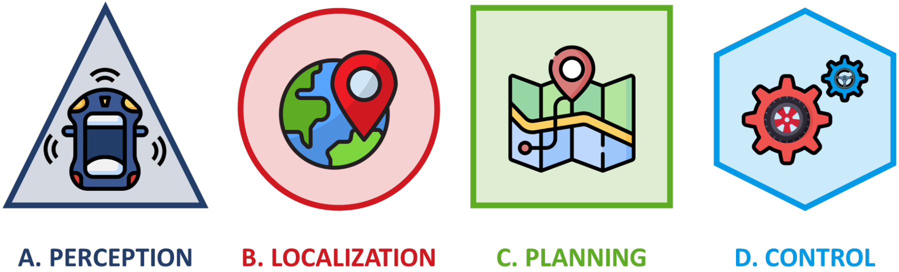
[1]Institute of Electronics, Computer and Telecommunication Engineering, National Research Council of Italy, Torino, Italy
[2]Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy

6th IEEE Conference on Control Technology and Applications
Workshop: **Control Systems and Robotics in the framework of Agriculture 4.0**

Autonomous agricultural vehicles

Autonomous vehicles in agriculture

- Provide favourable improvements to in-field operations;
- Extend crop scouting to large areas
- Perform in-field tasks in a *timely* and *effective* way.

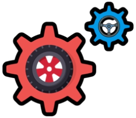


Guidance, Navigation and Control

- **Navigation** refers to the determination, at a given time, of the vehicle's **state vector**, exploiting filtering algorithms and sensors measurements.

- **Guidance** refers to the determination of the desired **trajectory** from the vehicle's current location to a designated target, as well as desired changes in velocity, rotation and acceleration for following that path.

- **Control** refers to the **manipulation** of the forces, by way of steering controls, thrusters, etc., needed to execute *guidance commands* while maintaining *vehicle stability*.



Mission framework

Mission scenario

- a Nebbiolo vine variety vineyard in Barolo;
- extending on a slopped terrain of about 0.7 ha;
- elevation range: from 460 m to 490 m a.m.s.l.;
- vertical shoot position trellis system;
- inter-plant/inter-row space: 0.9 m/2.5 m.

Mission framework

Autonomous vehicles

- (a) a fixed-wing UAV to collect information and aerial imagery;
- (b) a four wheel-steering electric UGV for in-field operations;
- (c) a mini quadrotor UAV for precision scouting above and within rows.

(a) MH900 by MAVTech

(b) e-AGRA by DiSAFA

(c) Q4T by MAVTech

Mission framework

Vehicles equipment: on-board sensors

(a) Taoglas Magma GPS

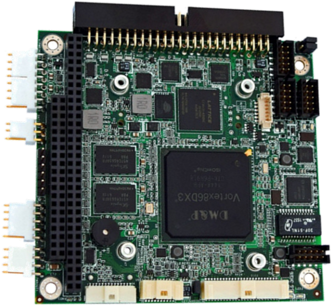
(b) Vectornav VN-200 IMU

(c) HC-SR04 US

Mission framework

Vehicles equipment: on-board computers

(a) Pixhawk 4 autopilot (PX4 FW)



(b) PC-104 OBC (RT-Linux OS)

Navigation

Navigation: Bayesian filtering

Bayesian filtering is a class of filters used as navigation system for autonomous vehicles.

- They leverage the **a-priori knowledge** of a dynamical system model **to estimate the state space** which maximises the **a-posteriori probability** of the **observations**.
- The estimation is performed through a **prediction-update** approach that effectively compensates for **noisy observations**.

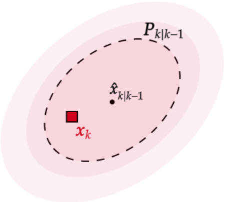
Bayesian approaches

1. **Kalman Filter (KF)**: pdf imposed to be **Gaussian** (i.e. $\mathcal{N}(\mu, \sigma)$).
2. **Particle Filter (PF)**: pdf **approximated** by a set of **weighted particles**.

Kalman filter

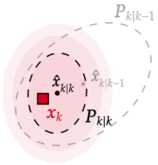
Two-step procedure

1. **Prediction phase**



Preliminary estimation of the system states based on: (*i*) *system model*, (*ii*) *applied control input*, (*iii*) *a-priori estimation*.

2. **Update phase**



Update of the preliminary estimation, computed on the base of the *current observations* (sensors measurements).

Distance filter

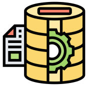
Enhancing navigation within crops due to:

1. **GPS data** typically neither reliable nor always available

   $\Longrightarrow$ **poor navigation data**.

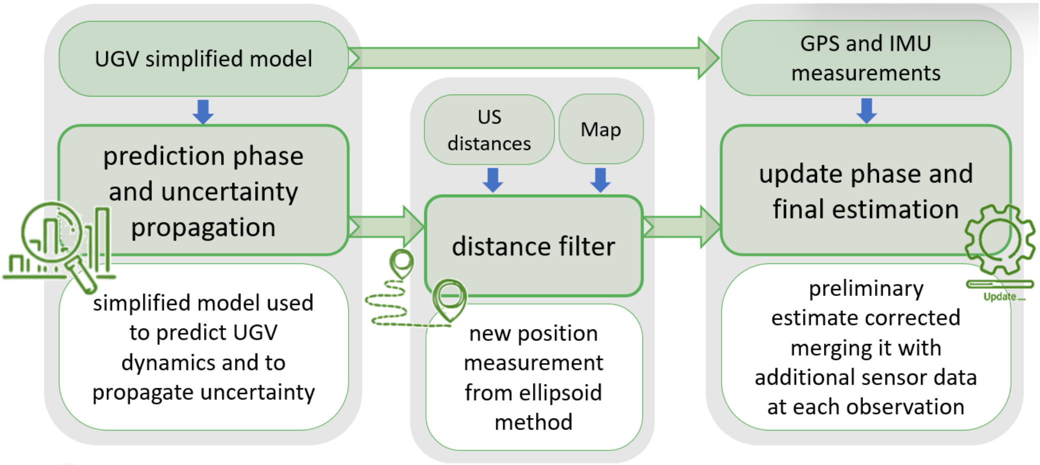
2. Valuable information provided by **3D digital maps**:
   - better comprehension of the **environment**;
   - data on **crops**, e.g. planting location, canopy shape, etc..



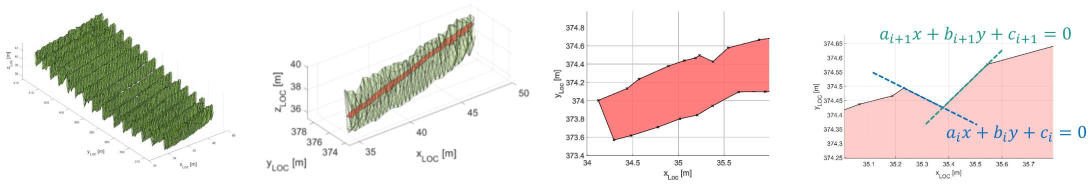



Proposed approach: Kalman-based distance filter integrating low complexity maps.

Distance filter



Row modeling



(a) (b) (c) (d)

⇒ the j-th **row** at given **height** h_{ref}, composed by N_j **segments** and described as

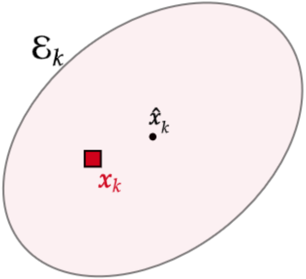
$$row_j = \begin{bmatrix} a_1 x + b_1 y + c_1 = 0 \\ \dots \\ a_i x + b_i y + c_i = 0 \\ \dots \\ a_{N_j} x + b_{N_j} y + c_{N_j} = 0 \end{bmatrix}$$

Ellipsoid method – Phase 1

Definition 2.1 (Confidence ellipsoid)

Given the the prediction $\hat{\boldsymbol{x}}_k$ and the covariance matrix of its uncertainty $\boldsymbol{P_k}$, the **confidence ellipsoid** $\mathcal{E}_k$, i.e. i.e. the **deterministic set** of possible **positions**, is defined as

$$\mathcal{E}_k = \{\boldsymbol{x} : ||(\boldsymbol{x} - \hat{\boldsymbol{x}}_k)||^2_{\boldsymbol{P}_k^{-1}} \leq 1\}$$

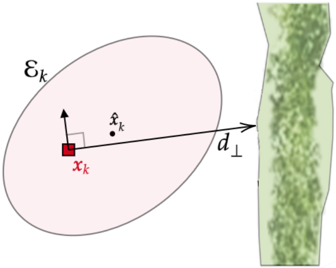


Ellipsoid method – Phase 2

Ultrasound sensors distance $d = d_\perp + e^d$ where:

- $d_\perp$: **measured** distance from the UV CoM to the map on the intercepted 2D slice;
- $e^d = e^s + e^m$: **unknown-but-bounded** error;

$$\implies d \in [\underline{d}, \overline{d}]$$



Definition 2.2 (Feasible point set)

Given the confidence ellipsoid $\mathcal{E}_k$ and the bounds on d, the **feasible point set** $\mathcal{F}_k$, i.e. the set of points which distance is coherent with the measured one, is

$$\mathcal{F}_k \doteq \{\mathbf{x} \in \mathcal{E}_k : \underline{d} \le d(\mathbf{x}) \le \overline{d}\}.$$

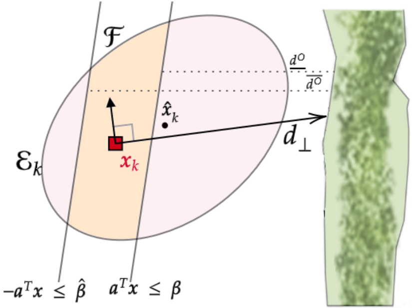
Ellipsoid method – Phase 3

Parallel cut method:

1. Identify the i-th segment compliant with d and $\hat{\boldsymbol{x}}$ $(a_i x + b_i y + c_i = 0)$;
2. Define the upper and lower offsets $\underline{d^O}$, $\overline{d^O}$ to find the parallel bounds of $\mathcal{F}$;
3. Identify the two lines, parallel to the one representing the i-the segment, i.e.

$$-\boldsymbol{a}^T \boldsymbol{x} \leq \hat{\beta}, \quad \boldsymbol{a}^T \boldsymbol{x} \leq \beta$$

where $\boldsymbol{a} = -\frac{a_i}{b_i}$, $\quad \hat{\beta} = -\frac{c_i + a_i \overline{d^O}}{b_i}$, $\quad \beta = -\frac{c_i + a_i \underline{d^O}}{b_i}$.



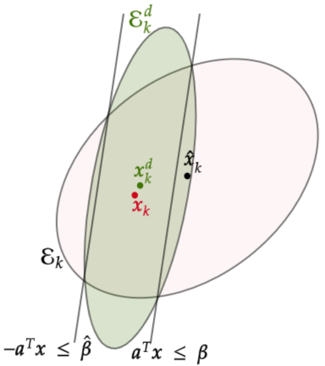
Ellipsoid method – Phase 4

Definition 2.3 (Propagation ellipsoid)

Given the confidence ellipsoid $\mathcal{E}_k$, defined by $\mathbf{x}_k$ and $\mathbf{P}_k$, and the geometry parameters $\mathbf{a}$, $\hat{\beta}$, β, compute the algebraic distance of each half-plane from the ellipse center, i.e. $\hat{\alpha}$ and α. The propagated ellipsoid $\mathcal{E}_k^d$ is defined by its center $\boldsymbol{x}_k^d$ and shape matrix $\boldsymbol{P}_k^d$, computed as

$$\boldsymbol{x}_k^d = \boldsymbol{x}_k - \tau \frac{\boldsymbol{P}_k a}{\sqrt{\boldsymbol{a}^T \boldsymbol{P}_k \boldsymbol{a}}}, \quad \boldsymbol{P}_k^d = \delta\left(\boldsymbol{P}_k - \sigma \frac{\boldsymbol{P}_k \boldsymbol{a} (\boldsymbol{P}_k \boldsymbol{a})^T}{\sqrt{\boldsymbol{a}^T \boldsymbol{P}_k \boldsymbol{a}}}\right),$$

where σ, τ, and δ are the **dilation**, **step**, and **expansion parameter** of the ellipsoid method, respectively.



Distance filter for UAVs

From 2D to 3D:

- presence of wind turbulence affecting the UAV attitude and altitude

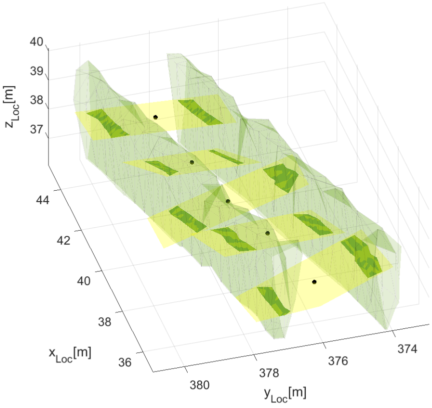
$\Downarrow$

real-time *roto-translation* of the reference plane containing UAV CoM;

- higher computational demand due to a larger configuration space

$\Downarrow$

introduction of a moving-window approach for accelerating reference slice selection.



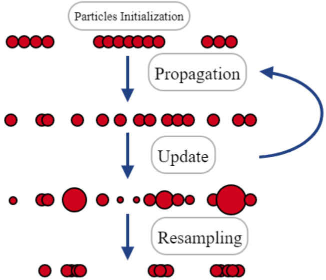
Particle filter

Kalman filter: 1 particle **VS** **Particle filter**: N_s particles.

Each particle is a possible realisation of state and provides its **estimation** and **reliability** (weight).

Three-step procedure:

1. **propagation phase** according to *system model*;
2. **weight update phase** according to *measurements*;
   higher weight ⇒ higher probability to be a *representative* sample;
3. **resampling phase** according to *updated weights*;
   higher weight ⇒ higher probability to be resampled *multiple times*.



Multiple weights particle filter

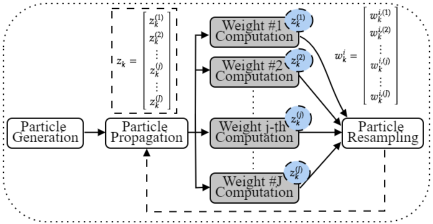
Particle filter for autonomous navigation of agricultural vehicles:

PROS: ability to deal with **non-Gaussian probabilities**.

CONS: high computational demand when applied to large systems and large N_s.

$\Longrightarrow$ **Multiple weights particle filter (MW-PF)**:

- system state space divided in J **partitions**;
- **multiple weights** associated to each particle;
- a **weight** for **each** partition;
- *more efficient* use of particles
- *more information* for each particle.



Single state weighted particle filter

When multiple, heterogeneous sensors are involved, observation features shall be included.

Proposed solution: single state weighted particle filter with distance filter.

The main features of the SSW-PF are:

a. each particle $\boldsymbol{s}_k^i$ is defined by the couple $\{\hat{\boldsymbol{x}}_k^i, \boldsymbol{w}_k^i\}$, where
   - $\hat{\boldsymbol{x}}_k^i \in \mathbb{R}^D$ is the i-th particle **state estimation**;
   - $\boldsymbol{w}_k^i = [w_k^{i,1}, w_k^{i,2}, \dots w_k^{i,j}, \dots w_k^{i,D}]^T \in \mathbb{R}^D$ is the **vector of weights** for i-th sample;
   - $w_k^{i,j}$ is the weight of the j-th state variable related to the i-th particle;
   - $\boldsymbol{w}_k^{(z)}$ is the weights updater vector from the sensor *(z)*, according to its observations.

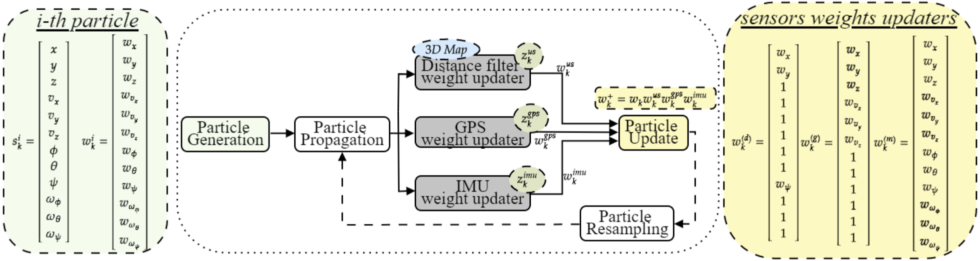
b. less particles required to achieve the **same** accuracy of a standard PF;

c. information carried by each particle **maximized**.

Multiple-weights particle filter

Proposed approach:

1. standard **propagation phase**;
2. **state-oriented weights update**: **weights updater** from each sensor observation, 1 for non-observed states;
3. **parallel, state-oriented resampling**: the single **state variables** are resampled.



Results

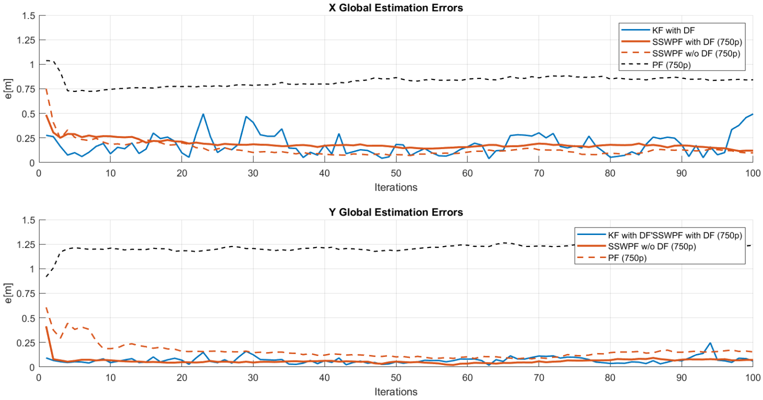


Figure 1: Estimation error for KF, PF, SSW-PF, and SSW-PF with distance filter.

Results

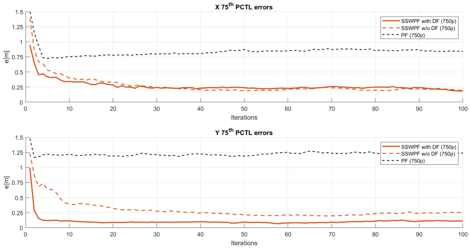


Figure 2: 75th percentile error for PF, SSW-PF, and SSW-PF with distance filter.

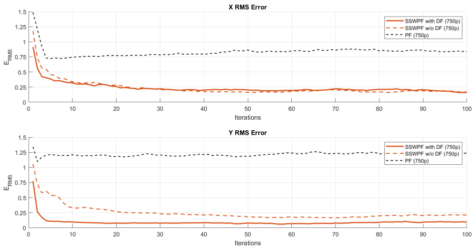


Figure 3: RMSE for PF, SSW-PF, and SSW-PF with distance filter.

Comparison of navigation algorithms

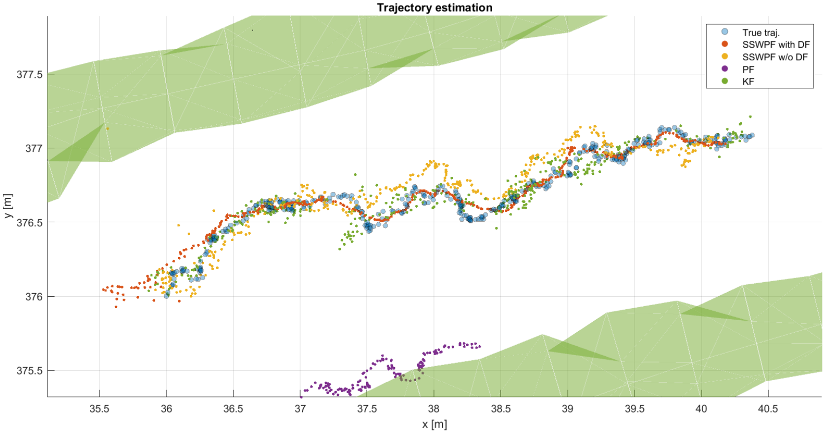
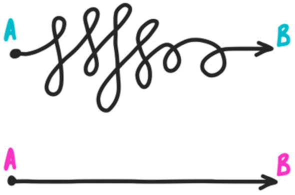


Figure 4: Estimated trajectory obtained using different approaches.

Guidance

Guidance for autonomous navigation

- We have to guarantee the ability to generate *optimal* and *feasible* path given:
  1. current vehicle location;
  2. mission and operative tasks;
  3. kinematic/dynamic constraints.
- Several criteria for path generation:
  1. shortest distance;
  2. minimum energy/consumption;
  3. maximum area coverage;
  4. etc.

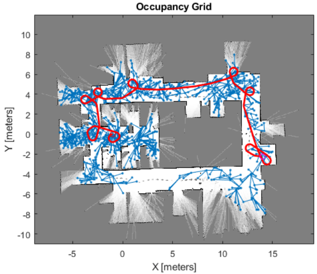


Motion planning

Definition (Motion planning problem)

Given a robot with d degrees-of-freedom in an environment with n obstacles, find a collision-free path connecting the current configuration (start) of the robot to the desired one (goal).

The robot and obstacle geometry are described either in a 2D or in a 3D workspace, while the motion is represented as a path in a (possibly higher-dimensional) configuration space.



Guidance for ground vehicles - Global planners

Global planners: generate intermediate goals (waypoints).

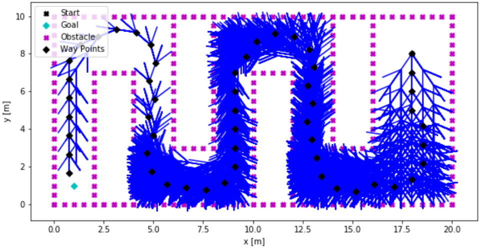
- **graph search-based schemes**, i.e. graph-search schemes computing paths over occupancy maps. Some examples are:
  1. *Dijkstra algorithm* (Madari, Adlonge, and Sharmila, 2019),
  2. A^* (Santos et al., 2019),
  3. D^* (Abrahão, Megda, Guerrero, and Becker, 2012).
- **sample-based path planners**, i.e. randomly sample the configuration space, looking for connectivity inside it and providing suboptimal trajectories. Some examples are:
  1. *probabilistic roadmaps* (Kavraki, Svestka, Latombe, and Overmars, 1996),
  2. *randomized potential fields* (Yan et al., 2020),
  3. *rapidly-exploring random trees* (LaValle, 1998),
  4. RRT^* (Messina, Fredda, Di Pietra, and Lingua, 2021).

Guidance for ground vehicles - Local planners

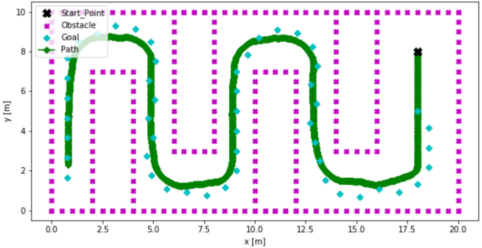
Local planners: guarantees smoothness and affordability.

- **interpolating curves**, often used as path smoothing solutions for a given set of waypoints. Some examples are:
  1. *line and circle curves* (Hsieh and Özguner, 2008),
  2. *clothoid curves* (Behringer and Müller, 1998),
  3. *splines* (McNaughton, Urmson, Dolan, and Lee, 2011),
  4. *Dubin (polynomial) curves* (Hameed, 2017).
- **numerical optimization planners**, i.e. minimize a given cost function subject to different constrained variables. The most important technique is:
  1. *dynamic window approach* (Guan, Tean, Oh, and Lee, 2019).

Proposed guidance approach for UGV



(a) RRT*



(b) DWA

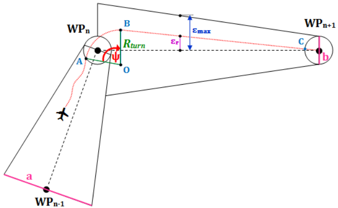
Guidance for aerial vehicles

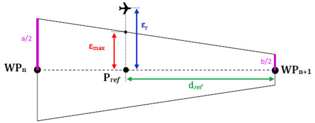
Different guidance algorithms depending on the type of mission the UAV was designed for (see Sujit, Saripalli, and Sousa, 2014; Rubí, Pérez, and Morcego, 2019; Quan et al., 2020):

- **direction field theory**: construction of a vector field that represents the desired ground track of the UAV, e.g. artificial potential field (Yingkun, 2018);
- **trajectory smoother**: transforming a waypoint-based path into a time-stamped kinematically and dynamically feasible trajectory (Capello, Guglieri, and Quagliotti, 2013);
- **informative path planning**: combination of global viewpoint selection and evolutionary optimization enforcing dynamical constraints (Popović et al., 2017).

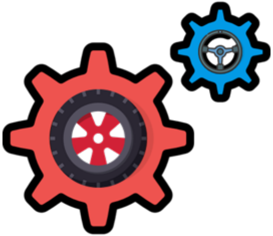
Proposed guidance approach for FW-UAV

- Defined a set of **2D** waypoints, we have:
  1. a *trajectory smoother*, to render the assigned trajectory kinematically feasible;
  2. a *cross-track error* ϵ_r as performance index for aerial mapping capabilities;
  3. a *look-ahead distance* for discerning two consecutive waypoints.
- Then, we add a **terrain following guidance**, based on a *ramp* function depending on the relative distance among UAV and *j*-th waypoint.





Control



Control for autonomous navigation

- Once the reference trajectory has been defined, either *offline* or *online*, it needs to be fed to the control block, which is in charge of *tracking the desired path* while eventually *fulfilling* operational, mechanical, and safety *constraints*.
- Several different control schemes have been proposed, tested and experimentally validated in the literature, also for agricultural machines, grouped into three main categories:
  1. **linear controllers**, e.g. PID, LQR, H_∞;
  2. **nonlinear controllers**, e.g. LPV, back-stepping, SMC, $\mathcal{L}_1$;
  3. **"intelligent" controllers**, e.g. fuzzy logic, NN-based.
- In the agricultural framework, (almost) all applications are based on PID and LQR since:
  - a. these algorithms are typically provided with the OBC/autopilot of commercial UVs;
  - b. they are simple to implement, easy to tune, and characterized by a very limited computational burden.

Control for ground vehicles

Some examples of control strategies for agricultural ground vehicles:

- **PID-based control** for effective weed and pest control (Gonzalez-de-Santos et al., 2017);
- **LQR** for a robot-trailer system with PSO (Wu, 2018);
- **SMC** for farm vehicles when subjected to sliding (Hao et al., 2004);
- **fuzzy control** for accurate inter-rows weeding (Li et al., 2020);
- **MPC** for autonomous navigation, path-tracking, and steering control.

Our approach, designed for a 4WS electric vehicle, aims at tracking the reference trajectory while minimizing the slippage generated by ASMs. Two-step approach:

1. *proportional steering control*, computing desired front/rear wheels steering angles;
2. *QP-based velocity optimizer*, enforcing non-holonomic constraints.

Control for aerial vehicles - How does MPC work?

MPC is like playing CHESS

- The choice of a move (*control action*) is realized by projecting in the future the game scenery (*dynamical process model*) and trying to predict how the opponent will answer to our moves (*output*).

- If in the next move the opponent answers in an unexpected way (*measurements*), we need to re-plan our move again in order to counteract the effect of the opponent move (*feedback*).

Control for aerial vehicles – LQMPC

- Let us consider a discrete-time, linear system $x_{k+1} = Ax_k + Bu_k, \quad x_k \in \mathbb{X},\ u_k \in \mathbb{U}$.
- The control problem is to minimize at each time k a given finite horizon cost function

$$J_T(x_k, \mathbf{u}_k) \doteq \sum_{\ell=0}^{T-1} \left(\|x_{\ell|k}\|_Q^2 + \|u_{\ell|k}\|_R^2 \right) + \|x_{T|k}\|_P^2.$$

- To solve the control problem, we *repeatedly* solve the following optimal control problem

$$\begin{aligned} \min_{\mathbf{u}_k} \ & J_T(x_k, \mathbf{u}_k) \\ s.t. \ & x_{\ell+1|k} = Ax_{\ell|k} + Bu_{\ell|k}, \ \ell \in [0, T-1] \\ & x_{\ell|k} \in \mathbb{X},\ u_{\ell|k} \in \mathbb{U}, \qquad \ell \in [0, T-1] \\ & x_{T|k} \in \mathbb{X}_T \end{aligned}$$

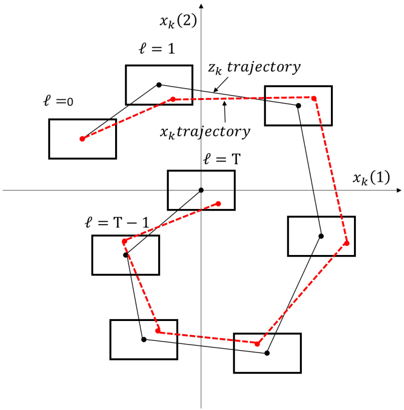
obtaining $\mathbf{u}_k^* = [u_{0|k}^*, \ldots, u_{T-1|k}^*]$ but implementing only the *first* control action $u_{0|k}^*$.

Control for aerial vehicles – TRMPC

- MPC performance degrades in the presence of uncertainty, leading to constraints violation and optimization infeasibility.
- Let's consider a discrete-time, linear system with bounded, additive disturbance $w_k \in \mathbb{W}$

$$x_{k+1} = Ax_k + Bu_k + w_k, \quad x_k \in \mathbb{X}, \, u_k \in \mathbb{U}.$$

- The objective is to control the associated nominal, undisturbed system subject to tightened constraints to allow all the trajectories to robustly lie in a tube centered on the nominal one.



Control for aerial vehicles – TRMPC

- We consider $x_{\ell|k} = z_{\ell|k} + e_{\ell|k}$, and $u_{\ell|k} = v_{\ell|k} + Ke_{\ell|k}$.
- Then, we design the *tightened* state and input constraints sets

$$\mathbb{Z} \doteq \mathbb{X} \ominus \mathbb{S}_K(\infty), \quad \mathbb{V} \doteq \mathbb{U} \ominus K\mathbb{S}_K(\infty), \quad \text{with } \mathbb{S}_K(\infty) \doteq \sum_{\ell=0}^{\infty}(A+BK)^{\ell}\mathbb{W}.$$

- The control problem becomes

$$\begin{aligned} \min_{\mathbf{v}_k} \; & J_T(z_k, \mathbf{v}_k) \\ s.t. \; & z_{\ell+1|k} = Az_{\ell|k} + Bv_{\ell|k}, \; \ell \in [0, T-1] \\ & z_{\ell|k} \in \mathbb{Z}, \; v_{\ell|k} \in \mathbb{V}, \qquad \ell \in [0, T-1] \\ & z_{T|k} \in \mathbb{Z}_T \end{aligned}$$

obtaining $\mathbf{v}_k^*$ but implementing only $v_{0|k}^*$ to obtain $u_k = v_{0|k}^* + K(x_k - z_k)$.

Control for aerial vehicles – SMPC

- Robust MPC leads to a pessimistic approach, too conservative when a safe level of constraints violation is allowed.
- Let us consider a system of the form

$$x_{k+1} = A(q)x_k + B(q)u_k + w_k.$$

- One solution is to adopt a probabilistic approach defining so-called chance constraints

$$\mathrm{Pr}_{\mathbb{W}}\{x_k \in \mathbb{X}\} \geq 1 - \varepsilon$$

and, selected $u_{\ell|k} = v_{\ell|k} + Kx_{\ell|k}$, we define a stochastic optimization problem to minimize

$$J_T(x_k, \mathbf{v}_k) \doteq \mathbb{E}\left\{ \sum_{\ell=0}^{T-1} \left(\|x_{\ell|k}\|_Q^2 + \|u_{\ell|k}\|_R^2 \right) + \|x_{T|k}\|_P^2 \right\}.$$

Control for aerial vehicles – OS-SMPC

- We propose a *sample-based* approach to design *offline* an inner approximation of the chance-constrained set restoring the results provided by the *statistical learning theory*.

Lemma 4.1 (Statistical learning theory bound)

Given $\delta \in (0,1)$ *and* $\varepsilon \in (0, 0.14)$, *if the number of samples* N *is such that* $N \geq N_{LT}$ *with*

$$N_{LT} \doteq \frac{4.1}{\varepsilon}\left(\ln \frac{21.64}{\delta} + 4.39 n_\theta \log_2 \frac{8 e n_\ell}{\varepsilon}\right) \tag{1}$$

then $Pr_{\mathbb{W}}\{\mathbb{X}_N \subseteq \mathbb{X}_\varepsilon\} \geq 1 - \delta$.

Control for aerial vehicles – OS-SMPC

OFFLINE STEP. *Before running the online control algorithm:*

1. *Compute the expected value* $\tilde{J}_T$ *of the cost function;*
2. *Draw* N *samples to determine* $\mathbb{X}_\ell^{S,\alpha}$, $\mathbb{U}_\ell^{S,\beta}$, *and* $\mathbb{X}_T^{S,\gamma}$*;*
3. *Remove redundant constraints and get* $\mathbb{D}$*;*
4. *Determine the first step constraint set* $\mathbb{D}_\mathbb{R}$*.*

ONLINE IMPLEMENTATION. *At each time step* k*:*

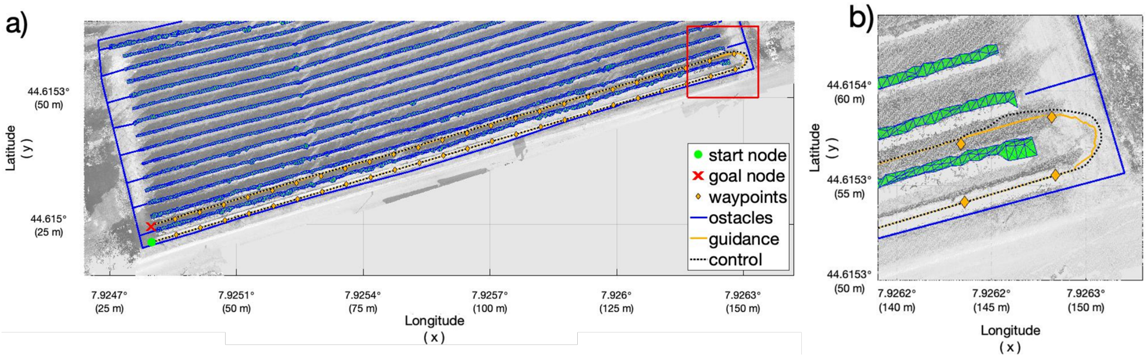
1. *Measure the current state* x_k*;*
2. *Determine the minimizer of the quadratic cost* $\tilde{J}_T$ *subject to* $\mathbb{D}$ *and* $\mathbb{D}_\mathbb{R}$

$$\mathbf{v}_k^* = \arg \min_{\mathbf{v}_k} \tilde{J}_T \tag{2a}$$

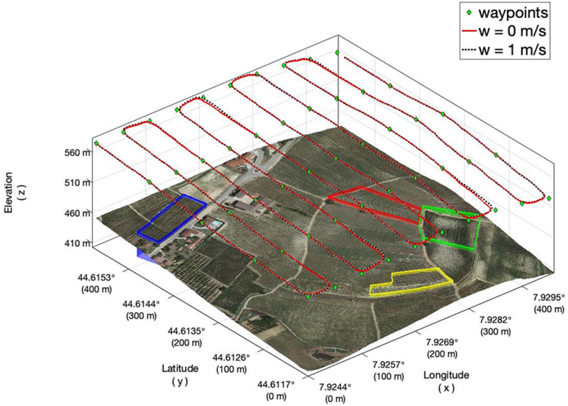
$$\text{s.t. } (x_k, \mathbf{v}_k) \in \mathbb{D} \cap \mathbb{D}_R; \tag{2b}$$

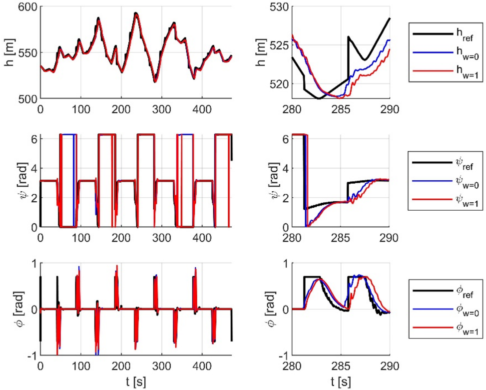
3. *Apply the control input* $u_k = Kx_k + v_{0|k}^*$*.*

Preliminary results for UGVs

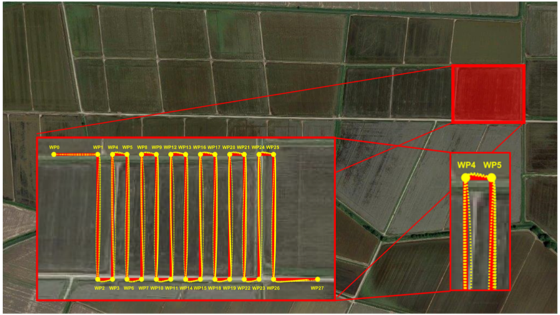


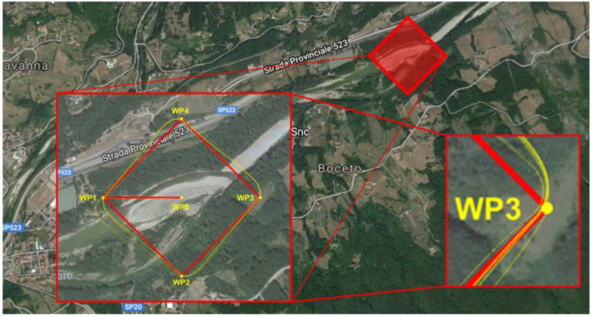
Preliminary results for FW-UAVs – TRMPC



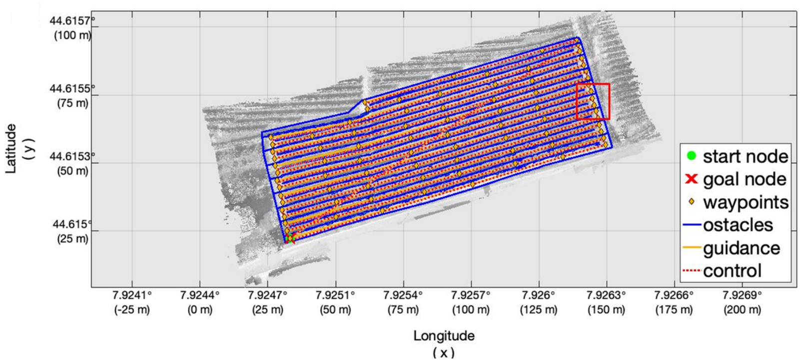


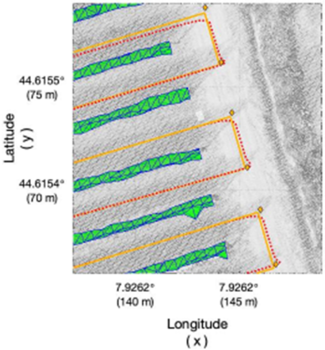
Preliminary results for FW-UAVs – OS-SMPC



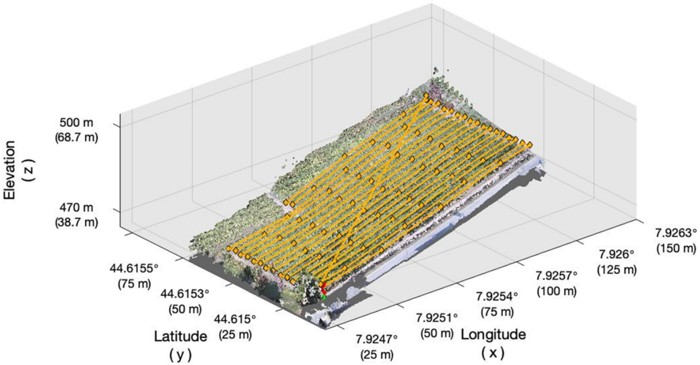


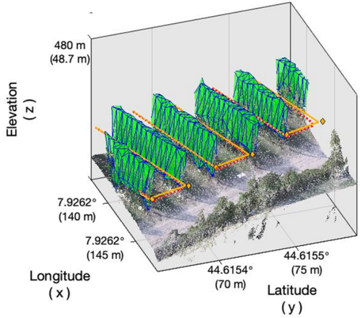
Preliminary results for RW-UAVs





Preliminary results for RW-UAVs





Thank you for your attention.

Q&A

cesare.donati@polito.it