XLIV Jornadas de Automática 2023

https://jautomatica.es

Implementing rover speed control in Paparazzi UAV

González-Calvin, A.[a,*], Jimenez, J.F.[1], García-Pérez, L.[1]

[a]*Departamento de Arquitectura de Computadores y Automática, Universidad Complutense de Madrid, Plaza de las Ciencias 1, 28040 Madrid, Spain.*

To cite this article: González-Calvin, A., Jimenez, J.F., García-Pérez, L. 2023. Implementing rover speed control in Paparazzi UAV.
XLIV Jornadas de Automática, 306-310. https://doi.org/10.17979/spudc.9788497498609.306

Abstract

This paper presents the design, implementation and testing of a speed controller for an autonomous rover with an unicycle drive. The aim is to get the rover to follow the speed setpoint as accurately as possible based on low cost GPS measurements. The use of IMU and GPS measurements was investigated and a moving average filter was designed to use the GPS signal as feedback for the speed controller. This filter and speed control were implemented on a flight controller board on a small autonomous rover using the the Paparazzi UAV development framework. Outdoor tests were performed.

Keywords: Filtering and Smoothing, Moving average filter, Speed control, Autonomous rover, Control of systems in vehicles, GPS

1. Introduction

Path planning of mobile robots is one of the basic operations needed to implement the navigation of the robot. The path planning operation provides the answer to the question 'how should I get to where I am going?' Tzafestas (2018). Answering this question involves, among other things, determining the direction and speed necessary for the robot to follow the desired path. This target speed and orientation is achieved at each instant by the acceleration to which the vehicle is subjected. This acceleration is translated into the force that the robot's actuators must exert at each instant. In fact, in many cases, the dynamics control cycle is assumed to be sufficiently fast and is omitted. In the case of a unicycle robot model, the input to the motion controller will be, on the one hand, the thrust and, on the other hand, the turning angle of the front wheels Francis and Maggiore (2016). The outputs will be the speed and orientation of the robot.

In many cases, path-following algorithms focus on the orientation that the robot must have at each instant of time in order to travel the target path, assuming that the velocity remains constant. This approach is valid in many cases, but in other cases it is necessary to control the speed of the robot. For example, if the robot has to follow paths with very sharp curves, it may be necessary to reduce the speed before entering the curve so that the robot can follow the curve accurately.

The most popular sensor for estimating the speed of a robot is odometry, as in Yeom (2020). Nevertheless, on steep, bumpy and stony terrain, errors in the encoder readings prevent a sufficiently accurate estimate of speed. Inertial units (IMU) can also be used to estimate the robot's speed, however IMU readings suffer from large amounts of noise. This is because the small, tracked robots, with a relatively small mass, get easily excited from small obstacles such as stones or sand, over which the robot drives, leading to big and frequent vibrations Shan et al. (2019).

In outdoor robots, GPS is a widely used sensor to estimate position and speed. However it is hard to obtain accurate vehicle speed relying on GPS for applications requiring real-time or high-accuracy speed estimations Yu et al. (2016).

In this paper, we present the current state of development of the LAB237 rovers, an experimental platform within which our developments related to autonomous outdoors navigation are integrated and tested. The LAB237 rovers have an IMU built into the autopilot and GPS but no odometry. Following section provides a description of LAB237 rovers and their equipment. Section 3 presents the speed control problem for our rover, the controller we have designed and section 4 the results with the real platform.

*alfredgo@ucm.es

2. Rover

The ISCAR (systems, control, automation and robotics engineering) research group currently has 3 similar small autonomous rovers. These rovers allow researchers and students to (1) easily test and debug the use of sensors, algorithms and strategies to be used in autonomous vehicles where experimentation is much more expensive and costly (such as autonomous surface vehicles); (2) perform experiments on multi-robot systems.

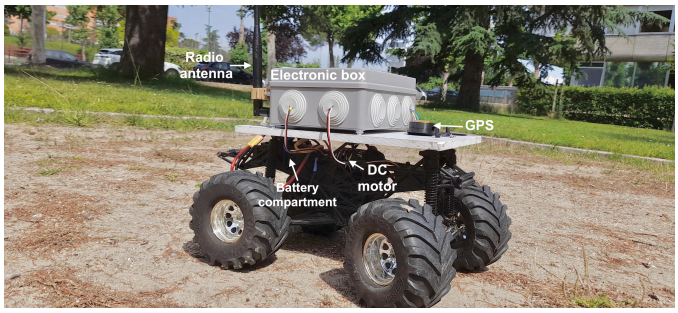


Figure 1: Rover

The rovers have been assembled using the chassis of a remote-controlled electric off-road car, Figure 1.

2.1. Actuator and sensor description

The rovers have an Ackerman type drive. Their movement is controlled by two motors: (1) a DC motor that provides thrust (2) a servo motor that drives the steering.

The on-board microcontroller is a Matek f765 flight controller or f405 in two versions, WSE or Wing. A flight controller is a circuit board equipped with sensors that detect vehicle movements and user commands. All flight controllers have basic sensors such as gyroscopes and accelerometers, while others may have other sensors such as air pressure sensors (barometer) and compasses (magnetometer). Other peripherals such as GPS, LEDs, servos, radio receivers and cameras can also be connected to the flight controller, Figure 2.

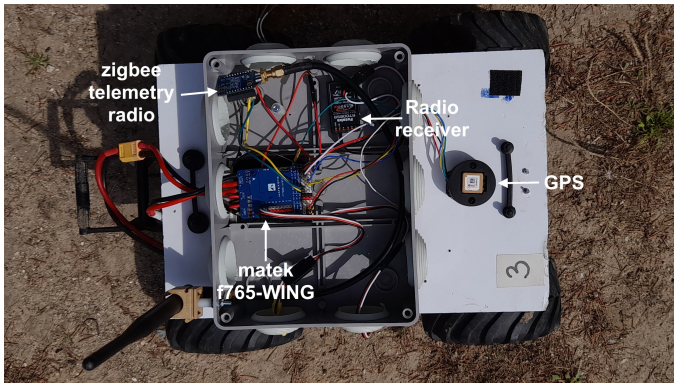


Figure 2: Rover hardware

Matek f765 and f405 flight controllers have an SMT32 microprocessor, the necessary electronics to power and control the motors, PWM outputs, a built-in IMU and numerous serial and digital inputs and outputs to facilitate the integration of various sensors. The rover used in the experiments shown in this work has an f405 flight controller.

In addition to the built-in IMU, the rovers have several connected devices: (1) an u-blox GPS (SAM-M8Q) receiver with built-in compass, (2) a wireless communication (Zigbee) transceiver that allows both monitoring of the rovers' parameters from the ground station and sending control signals to the rovers from the ground station, and (3) a radio receiver that allows the rovers to be remotely controlled using a radio used in aerial modelling.

2.2. Paparazzi UAV environment

We use Paparazzi as the programming and development environment for the autonomous vehicles. Paparazzi is a free and open-source hardware and software project intended to create a flexible autopilot system. One of the advantages of Paparazzi is the support for multiple hardware designs. In order to offer maximum flexibility and openness, the Paparazzi system was designed from start as a distributed one Gati (2013). Paparazzi has three levels of control loops. At the top, flight plan execution. At the middle level, the navigation controller, responsible for trajectory following. The lowest level is the controller that commands the servo actuators. Paparazzi also provides a simulator and a ground base station, Figure 3, where agents position are visualized and different sensor and internal variables can be monitored. Thanks to a modular software architecture, the process of generating the basic navigation code for the rovers has been accelerated by allowing the use of different modules available in the paparazzi framework, including GPS, radio control, wireless communication and others.

3. Speed control problem

In order to control the rover speed, we use as control signal u_v, the output of a feedforward + PI controller:

$$u_v = k_f v_{ref} + k_p e_v(t) + k_i \int_0^t e_v(\tau) d\tau \tag{1}$$

where v_{ref} is the desired speed, $e_v(t) = v_{ref} - v_{rover}$ is the error between the desired and the actual speed of the rover, and k_f, k_p, k_i are, respectively, the feedforward, proportional and integral control constants. The feedforward controller allows to reach a constant speed depending on v_{ref}, while the combination of the proportional and integral controllers acts in the presence of error, allowing the desired speed to be reached without steady state error. This control signal u_v is used to act directly on the throttle using the Paparazzi environment.

The need to measure the rover's speed, and the availability of sensors, allow us to choose between two perspectives. The first one, where the IMU can be used in conjunction with the GPS, as in Ding et al. (2020) where a multi-sensor fusion-based longitudinal vehicle speed estimator is proposed for four-wheel independently actuated electric vehicles for on-road electric car navigation. And the second one, where the actual speed is measured only with the GPS.

Since our rovers are equipped with an IMU, the first perspective could be appealing. However, several problems

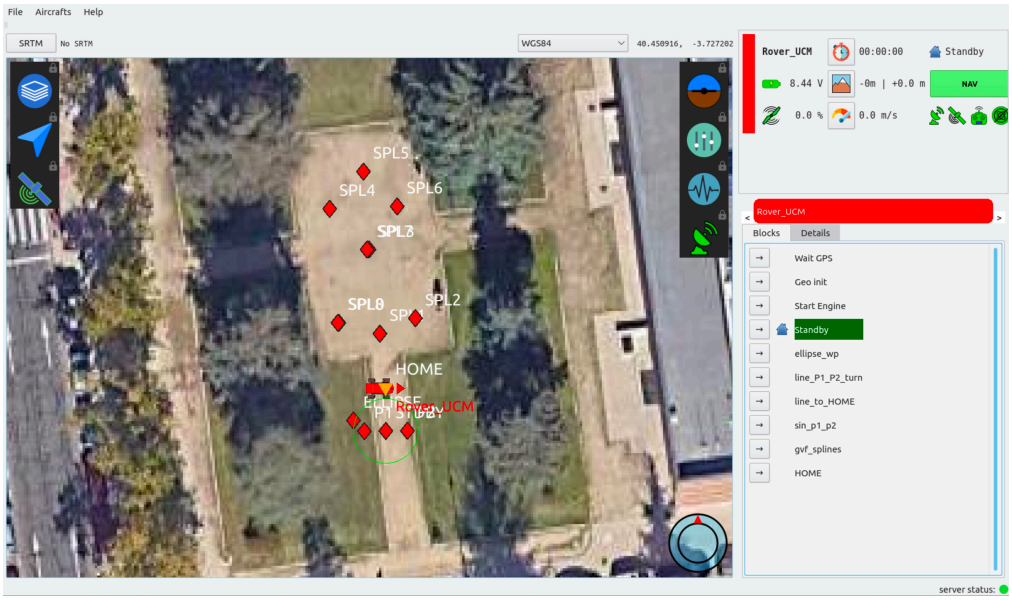


Figure 3: Paparazzi Ground Control Station

aroused when using this approach. Firstly, due to the bounciness of the rover, the accelerations measured with the IMU incorporated a big amount of noise and, since the speed is obtained integrating the acceleration, it also had noise. Secondly, performing filtering had no use, since the rover's acceleration transient time is faster than (or similar to) the noise perturbations. That is, the acceleration due to an actual increase or decrease in speed, could not be differentiated from the noise signal. This makes filtering and/or estimation unsuccessful, since filtration of the noise implies the filtering of the commanded signal as well. Consequently, this problems in combination with the GPS signal, created an oscillating speed reference signal, which could not be used in the controller.

Therefore, we measure the actual speed using only the GPS signal. When using low-cost GPS, the measured speed uncertainty increases as speed decreases. Since our rover moves at a relatively low pace, filtering is needed in order to reduce this uncertainty, and make it possible to use the measured speed signal in the feedforward+PI controller. This could be solved, in principle, using a Kalman filter. However, the process noise (the speed uncertainty) varies due to several factors involving the GPS. The three main GPS error sources are: (1) satellite orbit error and clock error at launch; (2) refraction error during signal propagation, which is unavoidable due to the non-uniform ionosphere and troposphere; and (3) clock error and noise during signal transmission. In complex urban environments, mountainous areas or areas with dense foliage, the GPS error can reach tens of meters Min et al. (2019). This factors create the necessity of another type of filtering, which must be: (1) fast enough with respect to the rover's speed, and (2) simple yet effective. These conditions can be achieved using a moving average filter as next subsection explains.

3.1. Speed control using moving average

As explained in the previous section, since the measured speed is directly taken from GPS, there exists and uncertainty associated with the measurement which depends on factors whose noise is difficult to predict, or even bound. In order to reduce this uncertainty and use the speed measurement signal directly from the GPS, a moving average filter is used in our rovers. The moving average is performed using the mean of the M previous speed measurement samples, that is:

$$v_{meas}[n] = \frac{1}{M}\sum_{k=0}^{M-1} V_{rover}[n-k] \tag{2}$$

However, (2) implies the computation of the mean of M samples in the time instants in which the speed must be measured. For relatively big M, this could imply a big delay between the speed measurement and the control action calculation. This delay may create situations in which the speed controller is not fast enough. For this reason, when implementing this filter, it is done in a modular fashion. That is, if M samples are taken and stored in the vector $V_{rover} \in \mathcal{R}^M$, the v_{meas} sample used as the measured speed value, can be computed with Algorithm 1.

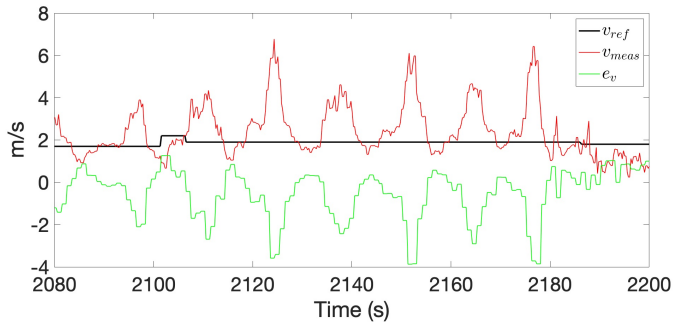
Algorithm 1 Moving average speed algorithm

▷ Initialize moving average array
$\mathbf{V}_{rover} \leftarrow \mathbf{0}$
$p \leftarrow 0$
$v_{meas} \leftarrow$ measure_speed()
while $p < M$ **do**
    $V_{rover}[p] \leftarrow v_{meas}$
    $p \leftarrow (p+1)$
end while
$p \leftarrow 0$
▷ Compute mean and get control action
while True **do**
    $v_{meas} \leftarrow v_{meas} - V_{rover}[p]/M$
    $V_{rover}[p] \leftarrow$ measure_speed()
    $v_{meas} \leftarrow v_{meas} + V_{rover}[p]/M$
    $p \leftarrow (p+1) \bmod (M-1)$
    $u_v \leftarrow$ compute_control(k_f, k_p, k_i, v_{meas}, v_{ref})
end while

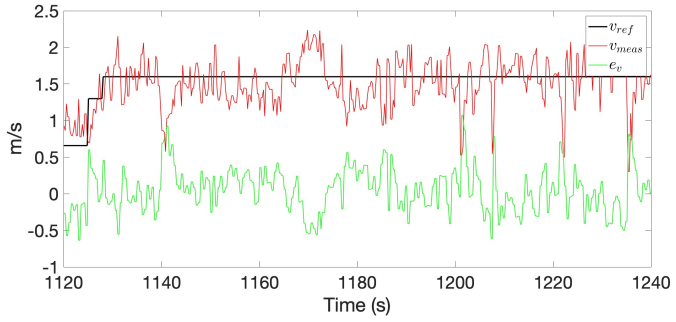
where *measure_speed()* measures the speed directly from the GPS, while *compute_control()* computes the control action u_v using (1).

4. Experimental results

In order to test this approach one of our rovers is chosen. For this purpose the rover speed setpoint (v_{ref}) has a lower bound limit of 0.2 m/s in order to deal with GPS noise and guidance control, which depends on $1/v_{rover}$. The speed is obtained from an u-blox GPS receiver with a working frequency of 1 Hz, and the moving average buffer consists on $M = 10$ samples. The experiment is carried by defining trajectories that the rover must follow with: (1) constant speed, independently of the curvature and/or the perturbations (i.e., terrain changes and GPS uncertainty); and (2) curvature varying speed set points, in order to test the controller operation.



(a) Speed computed using inertial sensors.



(b) Speed computed using GPS without moving average filter.

Figure 4: Speed measurements when following a straight line.

The speed algorithm 1, has been programmed in Paparazzi and it can be easily integrated and combined with other systems (i.e., guidance, obstacle detection, etc.). For more details about the implementation, the source code[1] is accessible in our Paparazzi repository, where it is combined with a guiding and navigation controller.

4.1. Results

Two scenarios are considered to study the implementation. A first scenario with a preliminary experiment to assess whether the IMU and GPS measurements alone are sufficient to provide feedback to the speed controller. In this first scenario, shown in Figure 4, the rover has to follow a straight line at a constant speed.

In the second scenario the aim is to answer the question of whether filtering the GPS measurements by a moving average is sufficient to feed back to the speed controller. In this case, shown in Figure 5, the rover has to follow an 8-shaped trajectory (made out of natural cubic splines), and the speed decreases if the rover is not aligned with the trajectory and/or the curvature of the spline increases. This means that in the second case the speed setpoint changes over time.

By analysing the results of the first experiment, shown in Figure 4(a), we can answer the question posed: the speed given by the INS (i.e., integrated from the acceleration measured by the IMU), cannot be used in the controller (1) because the speed oscillations are three or four times greater than the desired speed. Moreover, using only GPS measurements (shown in Figure 4(b)) improves the measurement, but the magnitude of the oscillations of the measured speed does not allow for a proper speed controller. Nevertheless, note that the vertical scale of Figure 4(b) is different from that of 4(a), so this approach is better than using the IMU alone. It is therefore necessary to implement the moving average filter proposed in this work.

Finally, in order to test whether the speed estimate using the moving average filter over the GPS signal was sufficient to feed back to the speed controller, the rover was programmed to follow an 8-shape trajectory with a time-varying speed setpoint. The experiments can be seen in Figure 5. It can be seen that even if the speed setpoint changes over time, the measured speed is good enough to allow the speed controller to work properly. This can be seen, in addition to the speed measurements (top figure), because the throttle (middle figure), as well as the integral and control actions (bottom figure), change according to the speed error, which is the difference between the setpoint and the rover's speed. It is important to add that on the day the splines experiments were carried out, the GPS signal and satellite coverage was not so good. Particularly, the position GPS uncertainty was 3.9 m for experiments shown in Figure 5 and 2.8 m in Figure 4. This shows that the approach used in this paper allows the speed controller to operate even when the GPS signal is not the best.

5. Conclusions

This paper presents the implementation of a rover speed control in the Paparazzi UAV. Through a series of experimental results, it has been shown that a low-cost GPS and a simple but effective filtering (implemented by a moving average filter) allow for a correct speed measurement that can be used as an input to a traditional speed controller. Moreover, thanks to the open source framework Paparazzi, the modularity of the implementation facilitates the integration of other systems with this speed controller. C code is shared on Github.

Future work includes incorporating sensors that allow the rovers to estimate their position and speed in the absence of GPS measurements, such as odometry or visual localisation.

[1] `https://github.com/UCM-237/paparazzi/tree/devel`

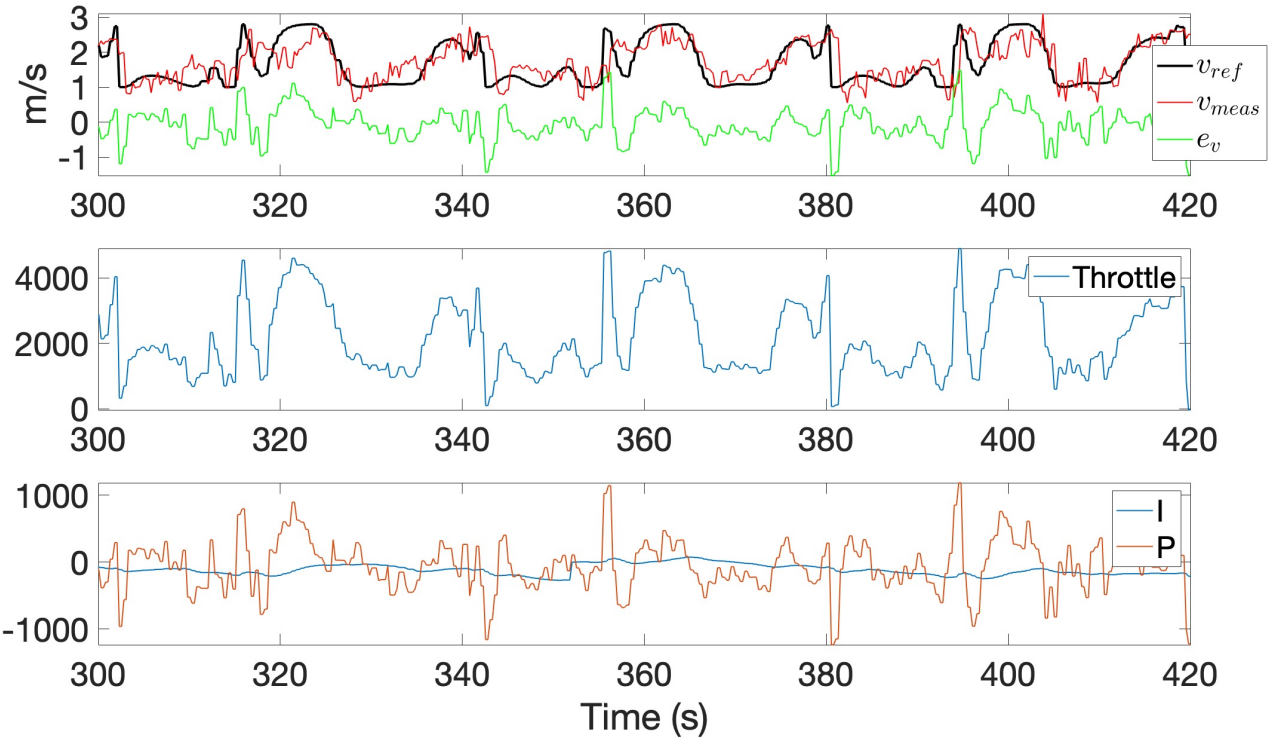


Figure 5: Speed measurements and control signals, when following splines and speed is adjusted in function of curvature, using a moving average filter.

References

Ding, X., Wang, Z., Zhang, L., Wang, C., 2020. Longitudinal vehicle speed estimation for four-wheel-independently-actuated electric vehicles based on multi-sensor fusion. IEEE Transactions on Vehicular Technology 69 (11), 12797–12806.

Francis, B. A., Maggiore, M., 2016. Flocking and rendezvous in distributed robotics. Springer.

Gati, B., 2013. Open source autopilot for academic research-the paparazzi system. In: 2013 American Control Conference. IEEE, pp. 1478–1481.

Min, H., Wu, X., Cheng, C., Zhao, X., 2019. Kinematic and dynamic vehicle model-assisted global positioning method for autonomous vehicles with low-cost gps/camera/in-vehicle sensors. Sensors 19 (24).
URL: https://www.mdpi.com/1424-8220/19/24/5430
DOI: 10.3390/s19245430

Shan, Z., Li, R., Schwertfeger, S., 2019. Rgbd-inertial trajectory estimation and mapping for ground robots. Sensors 19 (10).
URL: https://www.mdpi.com/1424-8220/19/10/2251
DOI: 10.3390/s19102251

Tzafestas, S. G., 7 2018. Mobile robot control and navigation: A global overview. Journal of Intelligent and Robotic Systems: Theory and Applications 91, 35–58.
DOI: 10.1007/s10846-018-0805-9

Yeom, K., 7 2020. Kinematic and dynamic controller design for autonomous driving of car-like mobile robot. International Journal of Mechanical Engineering and Robotics Research 9, 1058–1064.
DOI: 10.18178/ijmerr.9.7.1058-1064

Yu, J., Zhu, H., Han, H., Chen, Y. J., Yang, J., Zhu, Y., Chen, Z., Xue, G., Li, M., 1 2016. Senspeed: Sensing driving conditions to estimate vehicle speed in urban environments. IEEE Transactions on Mobile Computing 15, 202–216.
DOI: 10.1109/TMC.2015.2411270

Acknowledgments

This work is partially supported by IA-GES-BLOOM-CM (Y2020/TCS-6420) of the Synergic Projects programme of the Autonomous Community of Madrid, and INSERTION (PID2021-27648OB-C33) of the Knowledge Generation programme of the Ministry of Science and Innovation. Gonzalez-Calvin's work is specifically supported by the Collaboration Scholarship Program of the Ministry of Education and Vocational Training of Spain