GNSS-based environmental context detection for navigation

Florent Feriol, Yoko Watanabe, Damien Vivet

GNSS-based environmental context detection for navigation

Florent FERIOL[1], Yoko WATANABE[2] and Damien VIVET[1]

***Abstract*— Environmental context detection is a topic of interest for the navigation community since it enables to build a context-adaptive solution. Indeed if the type of environment is known it is then possible to choose the proper data processing algorithm or to select the sensors to be used to dynamically adapt the navigation solution design itself. This paper proposes to build a supervised machine learning model which can robustly classify multiple contexts such as urban canyons, urban, trees and open-sky areas using GNSS data only. A training and test database have been built with four datasets acquired at different times in order to prove the relevance of the solution. These datasets are made available to the community for research purpose. The choices of features and classifier are also discussed and compared to others papers. Our solution achieved an average 82.40% of classification accuracy.**

I. INTRODUCTION

In the recent years, with the addition of new constellations, hardware improvements and the development of new robustification algorithms, Global Navigation Satellite System (GNSS) based positioning has become more efficient than ever. The main issue with the latter is that very often those algorithms are dedicated to mitigate a specific limitation of GNSS which have a tendency to appear in a specific environment (also called context). For example, multipaths mitigation and Doppler aiding algorithms are mainly used in urban areas, shadow matching is exclusively used in urban canyon and the modification of the coherent integration time is useful in indoor or under canopy (see [1] for more details). This notion of environmental context is therefore considered as a key of future navigation solution since it could help to choose a suitable algorithm and improve the general performance of the solution [2]. This context-adaptive idea can also be extended to the navigation filter design itself by, for example, swapping between lose and tight coupling depending on the situation. Besides, such context information could also be useful for a path planning task in robotic applications, in order to ensure the navigation safety by predicting possible degradations of the localization precision.

Different methods already exist to extract the environmental context but require sensors which are not commonly used for navigation applications (cellular, Wifi, magnetometer, light detector) leading to more power consumption, bulkiness and cost [3], [4]. Even if multiple visual based approaches for context detection exist, the use of LiDAR or cameras require a huge processing power that should be dedicated to the navigation task by itself. In our opinion, there is a lack of simple and efficient environmental context classifier using GNSS in a kinematic case. Several articles tried to characterize GNSS signals in various environment and showed an evolution of the degradations depending on the context [8]–[10]. Hence, it is natural to think of using those degradations as a context detector. As introduced in [1] the semantic contexts of interest are: urban canyons, urban areas, forest/trees and open-sky. Note that indoor is excluded from our context of interest due to the fact that in deep indoor situation there is not enough visible satellites to compute a position and therefore a lot of features are missing for the classification step. If indoor is a key context for an application, a binary indoor/outdoor classification can be performed before applying the outdoor context detection approach proposed in this paper.

This paper is structured as follows. Section II introduces some existing works on GNSS-based context detection in order to showcase their limitations and the improvements we want to bring. Next, section III presents the strategy used to build the training/testing dataset as well as the acquisition setup. In Section IV, the choice of features used for the classification is discussed. Classifier, pre-processing and post-processing of data are explained in section V. Finally, experiments and results are presented in section VI.

II. STATE OF THE ART

Even if environmental context detection is a well studied problem with visual sensor or multisensor fusion approaches, few works have focus on this task using GNSS data. In [5] both GPS and GLONASS constellations are used to classify the context between indoor, urban and open-sky thanks to a two step processing. First, a Hidden Markov Model (HMM) predicts the indoor/outdoor context probability from means and variances of C/N0 (Carrier to noise ratio) features. Second, if the HMM output is unclear, a Fuzzy Inference System is used to separate open-sky samples from urban ones. Regarding the database, a unique dynamic test trajectory recorded at 1Hz with a smartphone is used. It contains approximately 100 samples. Results mainly focus on the detection capability during transitions. Thus there is no certitude on the reliability of the method for different constellations or localization since only a single test dataset was used.

[6] tries to classify indoor and outdoor using machine learning (ML) algorithms. 36 statistical features are extracted from azimuth/elevation, C/N0, Position/Horizontal/Vertical Dilution of precision (P/H/V-DOP), number of visible satellites (NVS). The database is composed of 195,861 samples recorded at different times with diverse smartphones and

[1]F. FERIOL and D. VIVET are from ISAE-SUPAERO, University of Toulouse, France. florent.feriol@isae-supaero.fr damien.vivet@isae-supaero.fr

[2]Y. WATANABE is from ONERA (DTIS), Toulouse, France yoko.watanabe@onera.fr

meteorological conditions. Results with multiple classifiers are shown and the stacking one, composed of several models, is the most efficient one. They also used a HMM in a post-processing step in order to improve the performances. However, the database is not available making us unable to know what trajectory the user did and at which speed.

[11] also use smartphones for pedestrian application but goes further than the previous paper by classifying four contexts: deep indoor, shallow indoor, semi-outdoor and outdoor. To do so the authors used a Recurrent Neural Network (RNN) model with an input vector of 11 statistical features based on the number of visible satellite and the signal to noise ratio. They showed that the use of multiple constellations could improve the classification results. However the dynamic test dataset is again very limited (only 322 samples) and therefore results cannot be extended to other locations and times.

More recently [12] proposed a method based on a fusion of RNNs and Fully Convolutional Networks (FCN) to classify in real-time three road navigation contexts: urban canyon, trees and open sky. Note that tree class is a road bordered by trees. The presented results show good performance. Nevertheless, a lot of information is missing. First, there is no information on the receiver and the antenna used. Second, the exact structure of the features vector is not mentioned and is mixed with features extracted with RNNs (which need a 256*11 historical data matrix as input - representing 50 seconds of signal). The architecture and the hidden size of the proposed RNN is not provided. Lastly, no information on the learning phase, the training and testing database (training epoch, batch size) are shared making the reproduction of the work impossible. More over, the system is based on the analysis of 50 seconds of signal making it more suitable for static system or with slow movement than for real navigation cases.

Finally, [13] tries to recognize six urban contexts: urban canyon, semi-urban, suburb, viaduct-up, viaduct-down and boulevard (roads bordered by trees). The definition of those contexts seems unclear since some classes look very similar to others. The authors proposed to use a five dimensional vector as input of a Support-Vector Machine (SVM) model. The vector is made of the mean μ and standard deviation σ of signal strength attenuation $A_i(t)$ defined in (1), blockage coefficient α (2) , GDOP expansion ratio λ (3) and the strength fluctuation coefficient β which are all extracted from the GPS L1 band.

$$A_i(t) = P_i^{std}(\theta) - P_i(t) \tag{1}$$

where $P_i(t)$ is the measured signal strength of satellite i at time t, and $P_i^{std}(\theta)$ is the standard strength with no attenuation for each elevation θ.

$$\alpha(t) = 1 - \frac{N_{visible}(t)}{N_{all}(t)} \tag{2}$$

where $N_{visible}$ is the number of visible satellites and N_{all} is the number of all satellites for the same constellation in the sky which can be obtained from the almanac.

$$\lambda(t) = \frac{GDOP(t)}{GDOP^{std}(t)} \tag{3}$$

where $GDOP(t)$ is the value computed by the receiver and $GDOP^{std}(t)$ is the GDOP value calculated for all the satellites in the sky. Finally, the strength fluctuation coefficient β is the standard deviation of the strength attenuation (1) in 1 second. The training data have been acquired in Shanghai (100 000 samples) at 5 Hz using an Ublox-M8N placed on the top of a vehicle. The test dataset have been recorded in Nanjing with the same setup. Data are pre-processed thanks to a standardization in order to assure a similar scale for the different features. A one-vs-all SVM model is then trained using the Shanghai data. The last step is to perform a Bayesian filtering at the output of the SVM classifier in order to improve the results by avoiding the random miss-classifications. The results are impressive but the ground truth shared is not very precise and the database is not available. Also, in order to compute the feature vector it is mandatory to obtain a position at each epoch and to have access to the almanac which is not always possible.

In summary, there are only few works dealing with GNSS signal based context detection. Even if existing works seems to give good classification results, they are based on small private datasets often obtained from smartphones held by pedestrians. It is important to note the relevance of having a large dataset with different conditions since it is the only way to ensure a generalizable solution. Also, the database are built in major chinese cities with contexts that can really differ from those found in Europe. None of these works can be reproduced as there is an obvious lack of public GNSS signal database. Therefore, contributions of this paper are twofold: first to propose a new GNSS-based environmental context detection methodology, and second to provide to public a GNSS signal database with labeled contexts[1].

III. Dataset

In order to acquire the database, a Traxxas rover (Fig. 1) equipped with a Ublox EVK-M8T receiver and a NovAtel VEXXIS GNSS-804 antenna was used. Data are collected at 5Hz thanks to a ROS platform. Compared to previous works dealing at 1Hz in static or low speed situations, we choose to increase the recording frequency to be able to deal with high speed situations. Moreover, as proven in [11], using multi-constellations could improve the detection capability thus the receiver has been configured to track GPS and Galileo signals. Given the rover's average speed of 4km/h the dynamic platform model of the GNSS receiver is set to portable.

To build a database, four experiments have been conducted on the ISAE-SUPAERO campus in Toulouse (France) on different days/hours and under sunny/cloudy conditions. A summary of the tests can be found in Table I. The training database has only been labeled when the context was obvious

[1]The database used in this paper is available at: `https://gitlab.isae-supaero.fr/d.vivet/GNSS_based_context_detection` and to be completed with new experiments

Fig. 1: Our rover setup at ISAE-SUPAERO. The robot is equipped with GNSS receiver and cameras for context ground truth labelling and visual confirmation.

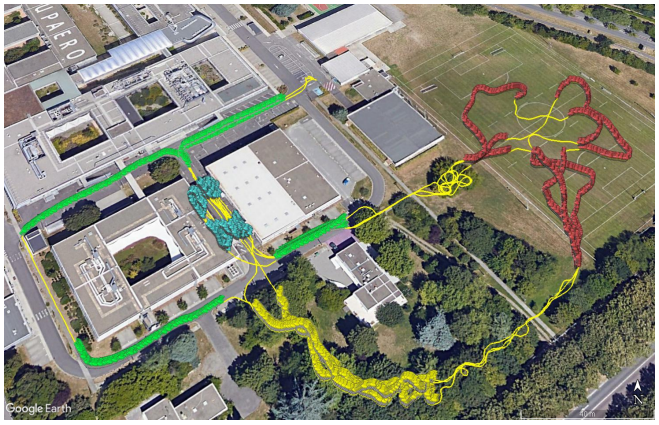


Fig. 2: Satellite view of the training database (:Urban, :Canyon, :Trees, :Open-Sky) and the ground truth trajectory in yellow

in order to have a generic model, thus every transition between contexts have been eliminated from the learning set. The ground truth labeling is based on the localization of the sample. Static points have also been filtered from training and test dataset as we are only interested in kinematic situations where the context can evolve. The number of samples in Table I is obtained after the filtering. Finally, to simulate a spatial independence for the tests some areas have been deleted for the training step (see Fig. 2). As a result there is no overlapped data point in training/testing subsets (Table VII).

IV. Features of Interest

In order for the proposed approach to be widely applicable, only the classical outputs which are accessible to end-users of standard off-the-shelf GNSS receiver are used to construct the feature vector. For instance, it excludes the use of correlators information. As using information from multiples constellations can improve the detection by reducing random outliers, features will be declined in their GPS and Galileo version at each epoch, except for the features based on the solution (such as the DOP feature).

Filename	Date	SoR	Meteo	Samples	
				Training	Testing
Dataset_1	05/28/2021	13h22	Sunny	4986	9292
Dataset_2	05/28/2021	16h49	Sunny	2529	5282
Dataset_3	05/31/2021	16h51	Sunny	2356	4240
Dataset_4	06/03/2021	11h45	Cloudy	4507	6684

TABLE I: List of the different recordings (SoR:Start of Recording)

The first feature selected is the number of visible satellites (NVS) as done in [6], [11], [13]. It gives a lot of information on the environment since its value will be very different in open-sky situations compared to urban canyon ones. However the NVS stays high in situations other than open-sky like under canopy, making this feature not sufficient to distinguish every context. In such areas the satellite signal quality should be impacted by multi-paths and/or attenuation due to branches and leaves. That is why a second feature based on the NVS is used but with a threshold on the C/N0 as done in [5]:

$$Mask(i) = \begin{cases} 1, & \text{if } C/N0_i > \xi \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

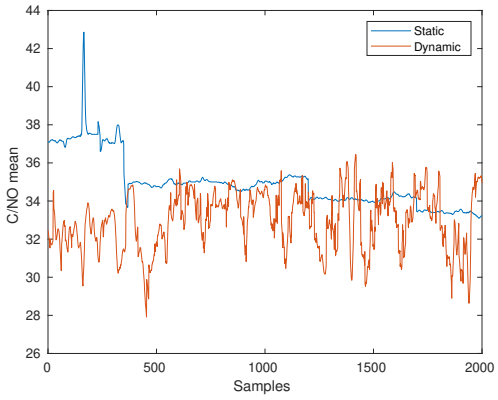
$$NVS\_filtered = \sum_{i \in \mathcal{S}_v} Mask(i) \tag{5}$$

with i the satellite ID, $\mathcal{S}_v$ a set of IDs of the visible satellites (i.e. $i \in \mathcal{S}_v$ if the i-th satellite is visible), and ξ the threshold value. In our case we found empirically that a proper value for the threshold was $30dB.Hz$. This threshold has been obtained by looking at the evolution of the signal quality when coming from an open-sky environment to under trees.

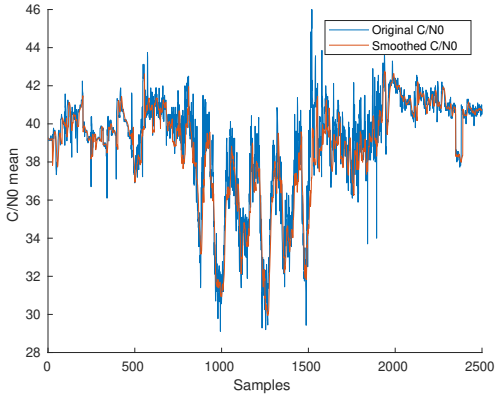
As mentioned before it seems mandatory to track the evolution of the C/N0 since it gives an idea on the signal quality. However in kinematic situation it is prone to a lot of variance (Fig. 3a). This problem is surprisingly not explained in the different papers that try to classify the context in real time [12] [13]. To resolve this matter we proposed to smooth the signal by using a filtering window on the last 10 samples (2s in our case) in which the mean and the variance are computed (6, 7) for each satellite (Fig. 3b). In addition, since an information on the whole constellation is needed, the mean of both the indicators is computed over all the satellites available (8, 9).

$$\mu_{C/N0_i}(t) = \frac{1}{10} \sum_{k=0}^{9} C/N0_i(t-k) \tag{6}$$

$$\sigma^2_{C/N0_i}(t) = \frac{1}{10} \sum_{k=0}^{9} \left(C/N0_i(t-k) - \mu_{C/N0_i}(t) \right)^2 \tag{7}$$



(a) Comparison of the C/N0 mean in a static and a dynamic situation



(b) Comparison of the C/N0 mean before and after smoothing

Fig. 3: C/N0 analysis in static and dynamic situation (a), and with the results of our proposed smoothing strategy (b)

$$\mu_{C/N0}(t) = \frac{1}{NVS(t)} \sum_{i \in \mathcal{S}_v} \mu_{C/N0_i}(t) \tag{8}$$

$$\sigma^2_{C/N0}(t) = \frac{1}{NVS(t)} \sum_{i \in \mathcal{S}_v} \sigma^2_{C/N0_i}(t) \tag{9}$$

After relying on the signal quality to extract information about the context, the receiver position solution could also give additional information. Indeed it is possible to estimate the error on the position which should be larger in cluttered environment. From the Ublox-M8T we can extract the horizontal/vertical accuracy and the position dilution of precision PDOP. For commercial reason, there is no access on how those values are calculated but they seem to be related to the position estimation error covariance. With the same idea, the pseudorange residuals which enlighten errors on pseudoranges (that can be due to multipaths among others) is also chosen as a feature:

$$res(t) = \sum_{i \in \mathcal{S}_v} |\rho_{m,i}(t) - \rho_{e,i}(t)|, \tag{10}$$

with ρ_m the measured pseudorange, ρ_e the estimated pseudorange. Here we decided to use the sum over the mean in order to avoid an over-smoothing of the values.

To have a better understanding of the constellation we can also use the elevation of each satellite as proposed in [6]. In cluttered environment the mean elevation will tend to be higher since tall structures will occlude the signal of low elevation satellites:

$$elev(t) = \frac{1}{NVS(t)} \sum_{i \in \mathcal{S}_v} e_i(t), \tag{11}$$

with e_i the elevation of the i-th satellite.
Finally, a 15 dimensional feature vector is obtained for each epoch:

$$\begin{aligned} v(t) = & [NVS_{gps}, NVS_{gal}, NVS\_filtered_{gps}, NVS\_filtered_{gal}, \\ & Hacc, Vacc, PDOP, \\ & \mu_{C/N0_{gps}}, \mu_{C/N0_{gal}}, \sigma^2_{C/N0_{gps}}, \sigma^2_{C/N0_{gal}}, \\ & elev_{gps}, elev_{gal}, res_{gps}, res_{gal}] \end{aligned} \tag{12}$$

V. Pre-processing, classifier and post-processing

A. Pre-processing

Once the feature vector is created, it still has to be pre-processed before using it as input of a classifier. To guarantee a similar scale for each feature, they need to be normalized since an SVM classifier will be used as justified later. To do so, the Z-score standardization (13) is used under an assumption of a normal distribution.

$$\bar{X} = \frac{X - \mu}{\sigma}, \tag{13}$$

To further improve the uniformity between small and high values, the sigmoid function define as follows is applied:

$$f(\bar{X}) = \frac{1}{1 + \exp^{-\bar{X}}}, \tag{14}$$

Those standardization steps are important due to the fact that SVM classifiers try to find the optimal hyperplane between two classes and therefore are sensible to magnitude differences.

At this point, we applied Principal Component Analysis (PCA) in order to verify the segmentability of our data (Fig. 4). It can be seen that except for the urban/open-sky case the different contexts are classifiable. It is important to mention that only the two main components are plotted on Fig. 4 and that the classifier will not be in this space. To compare the segmentability of our feature vector with the one proposed in [13], Fig. 5 shows the PCA plot for the feature vector which relies only on the five features listed earlier in Section II. It can be seen that the data are way more mixed up and that the resulting classification should be less efficient since 72.45% of the variance is represented here.

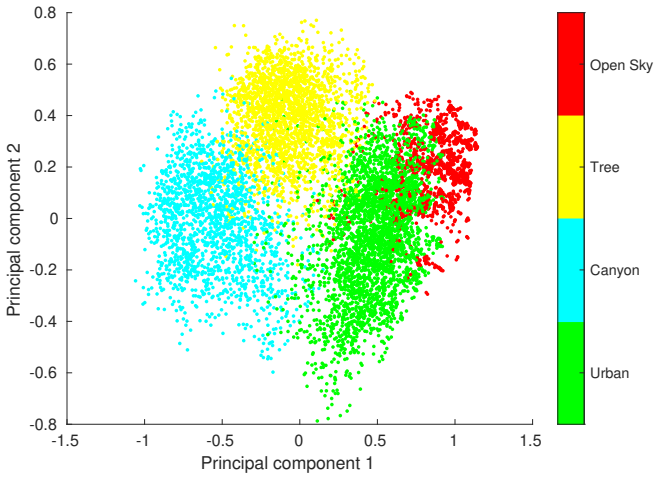


Fig. 4: PCA plot of the two main components (60.66 % of the variance) with our set of features

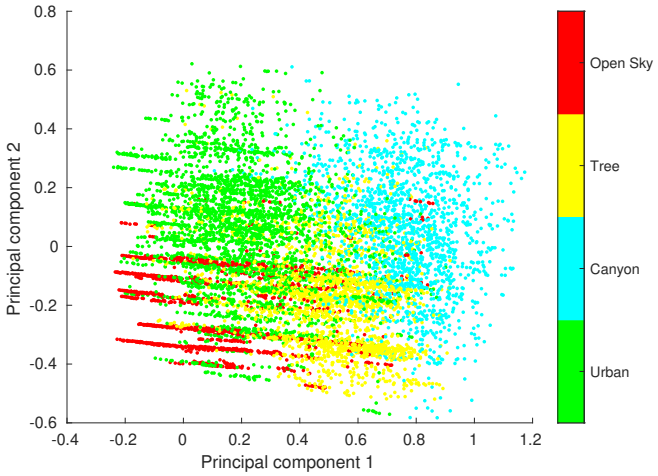


Fig. 5: PCA plot of the two main components (72.45 % of the variance) with the feature vector extracted from [13]

B. SVM classifier

As mentioned before, we chose to use an SVM classifier for the following reasons. The first is to avoid the randomness of algorithms such as boosting and bagging. Second, it allows to estimate posterior probabilities which can be useful for filtering/smoothing. Finally its fine tuning is more understandable than a black box such as deep learning solutions (SVM has only few hyperparameters). For multi-class problem, it has been shown that error-correcting output codes (ECOC) model can improve classification [7]. Indeed it reduces the possibility of wrong classification because each learner will find a single bit of the code whereas in other methods each learners define a class. Therefore this type of model will be used with SVM learners. The one-vs-one classifier is adopted due to the fact that it is less computationally-expensive than one-vs-all for training and performs better most of the time. We then obtain $K * (K-1)/2$ (K being the number of classes) binary learners and the coding matrix (Table. II) where 1 means that the class will be positive, -1 negative and 0 ignored during the training.

Class \ Learner	L1	L2	L3	L4	L5	L6
Canyon	1	1	1	0	0	0
Open-sky	-1	0	0	1	1	0
Trees	0	-1	0	-1	0	1
Urban	0	0	-1	0	-1	-1

TABLE II: Coding matrix of our ECOC model

It is possible to fine tune the type of kernel as well as the soft-margin and kernel scale hyper-parameter of the SVM learners. After multiple tests, it has been found that the gaussian kernel is the most efficient one with γ = 15.9631. The best soft margin parameter C is 4.0364. Those parameter have been found thanks to a Bayesian optimization.

C. Post-processing

Since the outputs of an SVM classifier are probabilities, it is possible to filter the result by applying a Bayesian approach. The prior probabilities are initialized uniformly for each context to 0.25 in order to avoid to favour a class over the others:

$$P(c_0|m_0) = [0.25, 0.25, 0.25, 0.25] \tag{15}$$

The posteriors of the next epoch can be computed using the previous probabilities as priors by Bayes' rule (16):

$$P(c_t|m_t) = \frac{P(m|c) * (P_{trans} \cdot P(c_{t-1}|m_{t-1}))}{P(m)}, \tag{16}$$

with $P(m|c)$ provided by the SVM, P_{trans} is the transition matrix between the contexts defined as follows (Canyon, Open-sky, Trees, Urban)(17). In order to define this transition matrix, we analyse the existing transitions in the dataset to get the transitions probabilities:

$$P_{trans} = \begin{bmatrix} 0.9 & 0.01 & 0.01 & 0.08 \\ 0.01 & 0.9 & 0.045 & 0.045 \\ 0.01 & 0.045 & 0.9 & 0.045 \\ 0.1/3 & 0.1/3 & 0.1/3 & 0.9 \end{bmatrix} \tag{17}$$

The context giving the highest probability of $P(c_t|m_t)$ is then considered as our predicted label.

VI. Experiments and Results

A. Setting and performance metrics

To show the relevance of our feature vector choice, different input vectors and pre-processing have been tried. Firstly, only GPS data are used in the input vector and the standard Z-score normalization (13) is the only pre-processing (SVM-G). Then the Galileo constellation has been added into the input vector as shown in (12) (SVM-GGa). In a third time the normalization is improved by the use of the sigmoid function (14) in addition (SVM-GGaS). Finally, the post processing Bayesian filtering presented in (16) is performed (SVM-GGaSB).The model has been trained using three of

the four trajectories and using the last one as a test. From each input vector we can extract the recall (18), the precision (19) and the F1-score (20) to demonstrate the performance of our solution. This evaluation summary is 4-folded.

$$Recall = \frac{True\ positive}{True\ positive + False\ negative} \tag{18}$$

$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \tag{19}$$

$$F1\text{-}score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{20}$$

B. Results

A comparison of the performances of the classifier for the four different input vectors is available in Table III where the best classifier is the one using GPS/Galileo features, sigmoid standardization and Bayesian filtering. The arctan function has also be tested instead of sigmoid but obtained inferior results. We also tried to fuse GPS and Galileo features as done in [12] but ended up with worse results. Because each constellation has its own characteristics it is better to keep their features independent. This result confirms the importance of using multiple constellations as mentioned in [11]. We also tested different training/testing datasets folding in order to validate the reliability and robustness of our solution with reference to: different constellation geometries (time of acquisition), different weathers. The F1-score for each test is available in Table IV.

Classifier	**Performances**		
SVM-G	R:0.8160	P:0.8194	F1:0.8177
SVM-GGa	R:0.8352	P:0.8303	F1:0.8328
SVM-GGaS	R:0.8338	P:0.8368	F1:0.8353
SVM-GGaSB	R:0.8407	P:0.8458	F1:0.8432

TABLE III: Average recall, precision and F1-score for each classifier (G:GPS, Ga:Galileo, S:Sigmoid, B:Bayesian filtering)

Classifier	**Test D_1**	**Test D_2**	**Test D_3**	**Test D_4**
SVM-G	0.8535	0.8568	0.8012	0.7662
SVM-GGa	0.8621	0.8458	0.8372	0.7903
SVM-GGaS	0.8516	0.8612	0.8385	0.8057
SVM-GGaSB	0.8625	0.8707	0.8410	0.8124

TABLE IV: F1-score for each classifier (G:GPS, Ga:Galileo, S:Sigmoid, B:Bayesian filtering) and each dataset

To see if the spatial distribution of the constellation has an impact on the classification results, the model is now trained using only the Dataset_1 and each of the remaining trajectories (Dataset_2, Dataset_3 and Dataset_4) are used as test (Table V). This table shows that F1-score is very close to those in Table III which is a promising result considering that a single dataset has been used to build the training database. This result shows that our solution is robust to a difference in time and that the constellation has a limited impact in a local environment, which is an unexpected result. Effectively there is a time difference of 1h30 between Dataset_1 & Dataset_2 and of 3h30 between Dataset_1 & Dataset_3\4. As a reminder the constellation period is about 12 hours. The same test has been conducted but with respectively the Dataset_2, Dataset_3 and Dataset_4 as training database and the remaining trajectories as test. The average F1-score is 0.8176 (respectively 0.8165, 0.8122 and 0.8240 for the three others training/testing configurations).

Classifier	**Performances**		
SVM-G	R:0.8253	P:0.7795	F1:0.8017
SVM-GGa	R:0.8481	P:0.8118	F1:0.8295
SVM-GGaS	R:0.8588	P:0.8218	F1:0.8358
SVM-GGaSB	R:0.8662	P:0.8218	F1:0.8434

TABLE V: Recall, precision and F1-score for each classifier (G:GPS, Ga:Galileo, S:Sigmoid, B:Bayesian filtering) using a single trajectory as training dataset

In order to check if a specific context is leading to larger number of wrong classifications, the confusion matrix for the SVM-GGaSB classifier is given in Table VI. It can be seen from this table that every context has a correct classification rate of at least 76.5%. The high classification rate of the canyon samples can be explained by its limited spatial distribution. To visualize where the errors are located, the predicted labels are plotted on a satellite view (Fig. 6). It can be seen that many missclassifications are coming from transitions or areas with a mix of contexts (such as trees close to buildings). Scores can also be plotted to have a better understanding of the Bayesian filtering (Fig. 7). Those values provide a confidence level of the context detection results, and hence could be very useful for decision-making of navigation filter adaptation or path planning application.

True Class \ Predicted Class	Canyon	Open-Sky	Trees	Urban	T%	F%
Canyon	2429	0	56	11	97.3	2.7
Open-Sky	0	2313	17	321	87.3	12.7
Trees	207	55	3246	292	85.4	14.6
Urban	590	454	661	5554	76.5	23.5

TABLE VI: Confusion matrix for the SVM-GGaSB classifier

As a final test, testing samples have been filtered depending on their position to prove that our solution is spatially generalizable. If a sample has a position at a distance of at least 3 meters from the closest point in the database the sample is kept otherwise it is filtered. We can then extract the confusion matrix built only on those samples (Table VII). This table shows that similar results than the classifier without filtering (Table VI) are achieved which demonstrates the spatial robustness of our solution.

VII. Conclusion

In this paper we proposed to used classical GNSS features provided by any off-the-shelves receiver and additional signal processing for context classification. We showed the possibility of using GNSS signals to extract the environmental context information with a mobile platform with an average F1-score of 0.8240 (with a single dataset as training). We also

True Class \ Predicted Class	Canyon	Open-Sky	Trees	Urban	T%	F%
Canyon	525	0	15	13	94.9	5.1
Open-Sky	0	1182	10	116	90.4	9.6
Trees	6	35	1848	263	85.9	14.1
Urban	450	439	414	3180	70.9	29.1

TABLE VII: Confusion matrix in the filtered areas

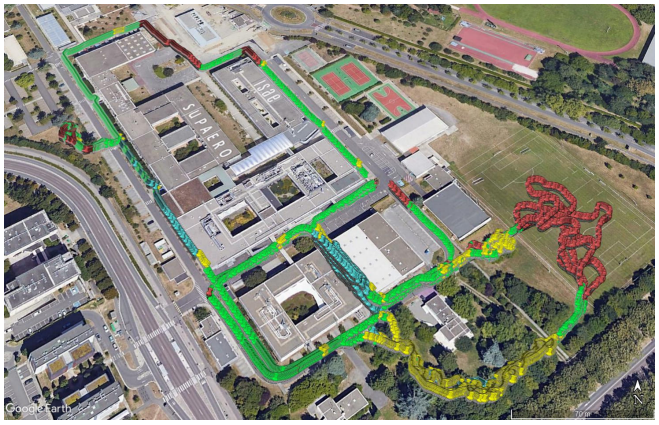


Fig. 6: Satellite view with the predicted labels (:Urban, :Canyon, :Trees, :Open-Sky)

showed that the methodology is robust to different constellation geometries and weathers. Considering the dataset, there is no GNSS dataset available in the communities making any comparison with state-of-the art solutions very complex. We decided to provide publicly this first set of experiments to the navigation research community. This dataset is going to be enriched and completed with on-coming experiments from bigger area and towns. In future works, we plan to provide additional synchronized sky-facing fisheye images. We think that the context detection capability can be improved with additional information coming from such sensor with few

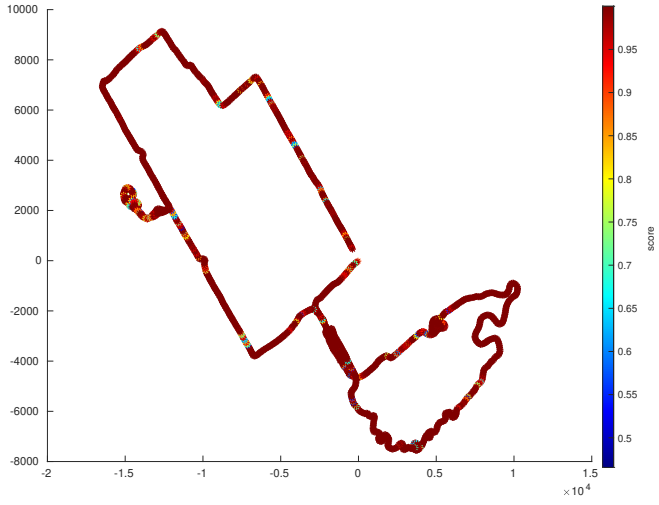


Fig. 7: Confidence score of the context prediction for each points of the tested trajectory

additional computing. For example, sky-segmentation could provide information on LOS (Line-of-sight) to each GNSS satellite. Our future work will focus on a fusion of such visual information with the GNSS-based ones to improve the proposed context detection solution.

ACKNOWLEDGMENT

This work was supported by the Defense Innovation Agency (AID) of the French Ministry of Defense (research project CONCORDE N° 2019 65 0090004707501).

REFERENCES

[1] F.Feriol and D.Vivet and Y.Watanabe, A Review of Environmental Context Detection for Navigation Based on Multiple Sensors, in Sensors, vol 20, 2020, DOI: 10.3390/s20164532.

[2] Groves, Paul and Martin, Henry and Voutsis, Kimon and Walter, Deborah and Wang, Lei, Context Detection, Categorization and Connectivity for Advanced Adaptive Integrated Navigation, in proceedings of the 26th International Technical Meeting of the Satellite Division of the Institute of Navigation, vol 2, 2013, ION GNSS.

[3] M. Ali and T. ElBatt and M. Youssef, SenseIO: Realistic Ubiquitous Indoor Outdoor Detection System Using Smartphones, in IEEE Sensors, vol 18, 2018, pp.3684-3693.

[4] Zhou, Pengfei and Zheng, Yuanqing and Li, Zhenjiang and Shen, Guobin, IODetector: A Generic Service for Indoor/Outdoor Detection, in ACM Transactions on Sensor Networks, vol 11, 2012, pp.361-362, DOI: 10.1145/2426656.2426709.

[5] Gao, Han and Groves, Paul, Environmental Context Detection for Adaptive Navigation using GNSS Measurements from a Smartphone, in Navigation, vol 65, 2018, DOI:10.1002/navi.221.

[6] Zhu, Yida and Luo, Haiyong and Wang, Qu and Zhao, Fang and Ning, Bokun and Ke, Qixue and Zhang, Chen, A Fast Indoor/Outdoor Transition Detection Algorithm Based on Machine Learning, in Sensors, vol 19, 2019, DOI: 10.3390/s19040786.

[7] T. G. Dietterich, G. Bakiri, Solving Multiclass Learning Problems via Error-Correcting Output Codes, in Journal of Artificial Intelligence Research, vol 2, pp.263-286, 1995, DOI: arXiv:cs/9501101v1.

[8] MacGougan, G.; Lachapelle, G.; Klukas, R.; Siu, K, Degraded GPS Signal Measurements With A Stand-Alone High Sensitivity Receiver, in Proceedings of the 2002 National Technical Meeting of The Institute of Navigation, San Diego, USA, 28-30 September 2002.

[9] Ma, C.; Jee, G.I.; MacGougan, G.; Lachapelle, G.; Bloebaum, S.; Cox, G.; Garin, L.; Shewfelt, J., Gps signal degradation modeling, in Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation, Salt Lake City, USA, 11-14 September 2001.

[10] Satyanarayana, Shashank, GNSS channel characterization and enhanced weak signal processing, PhD Thesis, University of Calgary, 2011

[11] Yan Xia and Shuguo Pan and Wang Gao and Baoguo Yu and Xingli Gan and Yue Zhao and Qing Zhao, Recurrent neural network based scenario recognition with Multi-constellation GNSS measurements on a smartphone, in Measurement, vol 153, pp. 107420, 2020, DOI: https://doi.org/10.1016/j.measurement.2019.107420.

[12] Liu, Haichun and Zhang, Minmin and Pei, Ling and Wang, Wei and Li, Lanzhen and Pan, Chang-Chun and Li, Zeya, Environment Classification for Global Navigation Satellite Systems Using Attention-Based Recurrent Neural Networks, in International Conference on Spatial Data and Intelligence, pp.60-71, 02-2021, DOI: 10.1007/978-3-030-69873-7_5.

[13] Wang, Yuze and Liu, Peilin and Liu, Qiang and Adeel, Muhammad and Qian, Jiuchao and Jin, Xiaoxi and Ying, Rendong, Urban environment recognition based on the GNSS signal characteristics, in Navigatio, vol 66, pp.211-215, 2019, DOI: https://doi.org/10.1002/navi.280.