Recognition of Surface Irregularities on Roads: a machine learning approach on 3D models

Rodrigo Huincalef[1], Guillermo Urrutia[1],
Gabriel Ingravallo[1], and Diego C. Martínez[2]

[1] Departamento de Informática Trelew, Facultad de Ingeniería,
Universidad Nacional de la Patagonia San Juan Bosco.
[2] Instituto de Ciencias e Ingeniería de la Computación, CONICET-UNS

Abstract. Roads are composed of various sorts of materials and with the constant use they expose different kinds of cracks or potholes. The aim of the current research is to present a novel automated classification method to be applied on these faults, which can be located on rigid pavement type. In order to collect proper representation of faults, a Kinect device was used, leading to three-dimensional point cloud structures. Images descriptors were used in order to establish the type of pothole and to get information regarding fault dimensions.

1 Introduction

The pavement of a street or highway is a structure composed of a set of layers of materials processed on the ground, whose function is to distribute the load of vehicles to the subsoil and allow the constant transit. The structure of the pavement should provide a surface of acceptable quality for the circulation of vehicles, adequate slip resistance, a lower level of noise, a waterproof, structural strength and a long life cycle with low maintenance cost. However, cracks are usual due to several factors. A *fissure* is a long, narrow opening in a slab of material that can be a corner fissure, longitudinal (if it extends along a slab) or transversal (if it extends perpendicular to the overturning of the slab material). The repair method for this type of failure consists of sealing joints and fissures, and repairing the entire thickness. On the other hand, a *pothole* is defined as a cavity, generally rounded due to the loss of the pavement on a part of the surface. The repair method for this type of failure depends on its deterioration, and is special for each case. In Figure 1 both faults are shown.

Fig. 1. A pothole and a longitudinal fissure, common in roads.

Due to the constant degradation of the different types of faults, these must be detected and repaired as soon as possible. The task of registering defects and depressions in the pavement has always been crucial to adopt a precise strategy for the maintenance and repair of roadways. However, manual measurement is a costly task both in time and resources. In this work we present an automated, AI-based classification of pavement faults, mainly potholes and fissures, through machine learning techniques using proper, accessible 3D models. There are previous works that apply different technologies for this purpose [2–12] but they require special devices and a good degree of important human intervention. Our interest is to formalize a process that is both accessible and cost-efficient. As a consequence, we implement an assistive application for the detection and classification of faults in the pavement, which will be useful for cost estimates. Our proposal make use of accessible domestic technologies, such as the 3D sensor of videogame consoles, to obtain a model of pavement fault that us appropriate for machine learning algorithms. A real implementation that offers an appropriate visualization and automatic geo-location of faults is also presented.

The work is organized as follows. Section 2 introduces the problem of road faults modelization. Section 2 explains the 3D sensors, the libraries to process the data obtained by these sensors, the management of point clouds, the descriptors that can be obtained from the analysis of these data and the machine learning technique used to generate the classification model. In section 3 we describe a prototype application for a vehicle that is used to collect the data through the Kinect sensor. Finally, conclusions and future work are discussed.

2 Surface Recognition

As stated before, we are interested in the automated classification of faults. This requires the use of adequate data structures representing real potholes and fissures. Hence, surface recognition is the first part of the process towards intelligent evaluation of faults. Ideally, the task of collecting data from real pavement should be also be automated, by using an autonomous vehicle. However, the construction of such a kind of vehicle is beyond the goals of our project. We are interested in the use of accessible hardware towards a practical solution for city governments and contractors.

In order to classify depressions then it is mandatory to get a proper mathematical model using some surface measure device. Here 3D sensors are used to analyse real world objects or environments and obtain their relevant physical properties, such as colors or shape, which later can be used to produce three-dimensional digital models. There are diverse, expensive sensors that may be used to attain this purpose, but we are interested in the use of domestic, accessible resources. Hence, we use a Kinect Microsoft sensor that has the capabability of generate Range Images, that keeps information regarding distance of each image pixel to the device capture point. We have used this device to record faults in a moving vehicle. The device should be mounted as shown in Figure 2, although that structure was not built during this research proyect.

The scanning process must produce 3D model representations. We use *point cloud structures*, which are point sets on a coordinate systems (optionally with RGB point information), being the most frecuently used the cartesian three-dimensional system (X,Y,Z). These representations can be renderized, inspectioned and converted to polygonal or triangular meshes.

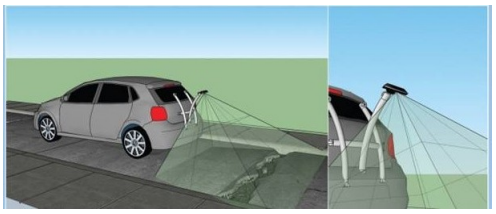

Fig. 2. The Kinect device can be easily installed on a vehicle.

There are several drivers and libraries (OpenNI, Freenect, OpenKinect,Point Cloud Library) that allow users to interact with the Microsoft Kinect sensor. Point Cloud Library (PCL) is an independent, open-source, multiplatform (also available in Windows, Linux, MacOS and AndroidOS), C++-written solution for sensing, geometrical point cloud processing and storing in 2D or 3D dimensions. The library PCL offers different standalone modules with algorithms that may be combined into a pipeline to spot several types of objects. These algorithms are meant to be applied on a wide range of tasks which is important for a correct object detection, for instance outliers filtering (filter point with values out of a certainrange), point cloud reading, storing, format conversion, decomposition (in order to perform searches) and concatenation, perform segmentation on specific parts of a complete point cloud capture, keypoint extraction and geometric descriptors computation with the aim of identifying distinct sorts of objects. In Figure 3 a point cloud of a real pavement fault is shown.

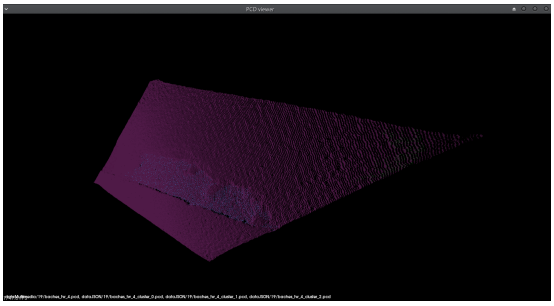

Fig. 3. Point cloud visualized with pcl_viewer tool from PCL. Full point cloud points are depicted with violet whereas points from a pavement depression, which have been isolated with classifier application, are with ligthblue and blue.

The process of obtaining a proper 3D model for automated classification is important and somehow difficult. Regarding the PCL pipeline for object recognition, it consists of a point cloud pre-processing phase with the goal of getting rid of noise. Then an object segmentation step is performed which allow us to obtain clusters which can be associated with potholes. Lasty, a descriptor generation phase is executed with the goal of getting information about the object geometry. PCL descriptors are divided in two classes: local and global. On the one hand, local descriptors describe surface geometry around a point without taking into account the complete geometry of the object the point belongs to. Hence when the local descriptors computation is performed, it is necessary to previously filter keypoints which belongs the object under study. Local descriptors applications include objects recognition and registration, which is a technique to detect whether common areas in several point clouds exists. On the other hand, global descriptors describe geometry of a whole point cluster which represents an object. Therefore if there is a need to generate this type of descriptor it would be compulsory to perform previously a pre-processing step in order to isolate the cluster from the point cloud. Applications of this type of descriptors are the object recognition and classification, position estimation and geometry surface analysis (type of object, shape, etc.).

The process of collecting samples took several days. Also, during this phase it was observed that some cracks does not possess significant depth to be measured by the Kinect device. Althoug this seems to be a drawback, those cracks are not of real interest for evaluation costs of repair (at least not for the moment, because faults tend to increase in size as time goes by). Due to this fact we decided to classify only those types of cracks which have enough depth to be isolated and described by descriptors which make use the of geometrical information regarding angles between normals of a surface. Because of this, descriptors were selected taking into account processing hardware capabilities and normal associated descriptor properties. These are the following:

- Global Radious-based Surface Descriptor (GRSD): This descriptor does not use any sort of pre-processing, for instance normals, yet it consists in traversing each of the point cloud points and for each select three random points and compute three ratio functions: D2, D2 ratio, D3 and A3. Every function produces histograms which describe the geometric relation among points that are part of the main figure.
- Ensamble Shape of Functions (ESF): Global descriptor that is based on Radio-based Surface Descriptor (RSD), which consists in geometrical surface shape by calculating radial information through neighbour points. Therefore this algorithm consist in setting up a formula using normals angle relation λ, the distance between them d and the surface radious r defined in $d = r * \alpha$. The latter formula is applied to every point p, that belongs to the point cloud, what gives as a result a set of ratios which describes each of the spheres that contains p with every one of the neighbours. Finally, only the minimal and maximal ratios are selected to be included in the final descriptor for that point.

The outcome of these complex processes is a data structure properly representing relevant pavement faults. There is, however, the need of determining whether these faults are either potholes or fissures, as they require different treatments. In the following section we discuss the use of machine learning techniques to help this task.

3 SVM

Support Vector Machine is a technique used for supervised learning to solve classification and regression problems based on hyperplanes. In this method the p features associated with a certain amount of training samples are arranged in a *p-dimensional* (being each sample a p dimensional vector) where each feature represents the coordinates with regard to one specific sample. In this way the SVM model generation is based on computing, testing and selecting from several hyperplanes which separate input data in order to get the best. Thus, SVMs allow researchers to divide data in sets with linear hyperplanes. Nonetheless, there are situations in which input data cannot be linearly divided, causing an unsatisfactory performance. The solution offered by SVMs to this problem is the use of kernel decision functions which convert the feature input data space to a higher dimension space being these quadratic, cubic, polynomial or higher dimensions space with the goal of achieving the finding of a brand-new hyperplane that separate samples with a better precision. Therefore there are a diverse range of kernels employed to attain this objective, mainly Linear, RBF and Polynomial.

As a result of SVM being a classification binary algorithm, various techniques have been developed to attain multiclass classification which allow researchers to assign samples among two or more classes. Two of these methods are:

- One versus One (One-vs-One, OvO): Given N classes this algorithm perform the $N(N-1)/2$ binary classifiers training getting each for every class combination using the training dataset. Then during the prediction phase a voting scheme where every classifier gives a result for the same input data sample. Hence the class which has a greater amount of positive votes it is the result of the prediction process.
- One versus Rest (One-vs-Rest, OvR, OvA): Given N classes N classifiers are trained using the full training dataset, choosing as positive those classes that belong to the classificator whilst the others are chosen as negatives. Accordingly, by receiving an input sample every classifier generate a decimal value which is a confidence score that indicates the probability of that sample of belonging to that type of element. Consequently the class with the highest score is considered the class in which the element is assigned.

In our current research project, an SVM implementation using C++ programming language was implemented for constructing the classification model (converting the data point cloud samples into SVM compatible format). Throughout this process, it was necessary to assign a class to every type of sample (crack

or pothole) using OvO technique and testing the same dataset with Linear and RBF kernel.

3.1 Building a classification model

The process of construction of the classification model consisted in the following steps:

1. Pavement surface samples collection: During this step an estimated amount of 1000 samples were captured using the Kinect sensor, which sensed different pavement streets in the city of Trelew.
2. Samples pre-processing phase: The main target of this phase is the preparation of the raw samples in order to improve quality. Several PCL algorithms for discarding noise were studied. Consequently they were tested against several samples. Next, the amount of point cloud points was reduced because of the hardware limitations and high samples quantity. Thus, *downsampling* techniques were implemented. Finally, it was necessary to obtain geometric properties in order to describe with higher precision the sensed pavement surface. Here, PCL normal computation algorithms were employed.
3. Sample segmentation: Algorithms for segmentation PCL were analyzed employing the samples previously captured with the aim of isolating point cloud clusters which belong to the depressions on the pavement and the rest of the street. As a consequence, Planar and Euclidean Segmentation algorithms were selected and concatenated in the instructions pipeline.
4. Curvatures computing: A deeper comprehension of fault dimension is obtained. The average curvature for clusters isolated earlier, which are thought to be part of a pavement depression, were computed so that allowed us to get curvatures minimum and maximum ranges, which let us get rid of those segments whose average curvature level was out of these ranges. Therefore this helped to improve the classifier application ability to filter or keep clusters based on alike geometry surface depressions.
5. Feature building: Lastly, diverse PCL surface description methods were studied and compared with regard to the size in bytes for each sample and amount of samples to be processed, histogram shape associated with different types of samples and sort of descriptor (local or global). Later it was investigated whether PCL offered machine learning models support combined with descriptors. Due to the fact the library had machine learning SVM support (making use of libsvm tool) this method was chosen for the classification application and each one of the point cloud samples were converted to libsvm compatible format.

After faults were detected by the sensor and raw data was obtained, a complex process of data preparation is triggered. Although it was an integral part of the project, it is not, however, the intention of this paper to describe the process in detail, other than the stages mentioned above. In the following section we address the automatic classification of faults.

4 Automatic classification

The first step toward the construction of the model was to assign 76 % of samples to training partition leaving the remaining for testing partition. Previously mentioned descriptors were tested in several experiments with various types of samples and as its precision was not enough it was decided to add depressions properties (for example length, width, depth and volume) along with the descriptors. Potholes properties were computed by PCL algorithms which are based on the minimum and maximum Cartesian axis values (AABB, OBB y "boundary cloud"). Thus statistical size depression values were compared as well as arithmetical relationships among them (averages, arithmetical difference between size and average sizes) with the goal of finding a way of make a difference among different types of depressions.

By observing these values it was seen that the width-height difference in the training dataset was similar for the same type of sample. Then a re-classification sample step was performed. This consisted in establishing a limit value to distinguish cracks from potholes being the samples that have a $|length - width| > 0.49$ difference value considered as cracks whereas others were considered as potholes, since cracks have considerably higher length than width. Thus the previous formula utilized these values for computing the difference using PCL Oriented Bounding Box (OBB) algorithm on X and Y axis.

Having observed the classification precision increased by reclassifying sample training dataset the same technique was applied to the testing dataset which allowed us to filter 806 out of 1000 samples (753 samples for training and 53 for testing). Afterwards new tests were performed using the (now) reclassified samples, obtaining 89 % accuracy with Linear kernel and 71 % with RBF (gamma 0.0000002 and cost C 1500) employing a five iteration cross validation approximation with GRSD descriptor. Next the length-width difference was re-tested, but with ESF descriptor. Using the same parameters, the process obtained a 98 % precision score for Linear kernel and 54 % for RBF kernel.

A comparison of classification metrics concerning the selected descriptors (using the original dataset partition with 50 samples) was made, with the goal of verifying the classification performance of ESF against others. These values can be found in Figure 4 and 5.

In the following section we describe the use of a vehicle in the collecting samples stage, as shown in Figure 2, and the embedded system for registering faults and their position by geo-localization.

	Kernel Linear			Kernel RBF		
Tipo de muestra	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Baches	1.0	1.0	1.0	0.0	0.0	0.0
Grietas	1.0	1.0	1.0	0.17	1.0	0.29
avg/total	1.0	1.0	1.0	0.03	0.17	0.05

Fig. 4. ESF descriptor metrics with Linear and RBF kernel.

	Kernel Linear			Kernel RBF		
Tipo de muestra	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Baches	0.83	1	0.91	0.00	0.00	0.00
Grietas	0.23	0.78	0.36	0.17	1.00	0.29
avg/total	0.80	0.53	0.58	0.03	0.17	0.05

Fig. 5. GRSD descriptor metrics with Linear and RBF kernel.

5 Application for a vehicle prototype

Ideally the sensing prototype device consists of a vehicle with a shell fixed in its back part, as it is seen in Figure 2, which allows the operators to hold the device from an appropriate high and take samples directly from above the pavement surface. The sensor is connected to an electrical supply device as well as a notebook in which the pavement depression measuring application is running. This application is set up to work with a GPS device that provides the fault location (latitude and longitude). Thus the pothole measuring process workflow is to stop the vehicle when a pavement depression is spotted, then the application user performs the measuring of it and the car moves to the following one and so forth. This process allows the sensing and registration of big areas of a city in acceptable time. The application main architecture is composed of the following software modules:

- Kinect device: It is the main sensing device which let the application get video and depth frames, which are compulsory to the point cloud file building. The frames are continuously read, rendered and visualized in real time from the depression capture application screen.
- Geofencing module: Its purpose it is to compute and return GPS coordinates in which a pavement depression is.
- APIClient: This component contains the main data exchange class between capture and web application.
- Client capture application: It offers the functionality with respect to capturing depression process and sending to the web application.

5.1 Pothole classification application

The pothole classification application is developed in C++ and its operation consists in reading configuration parameters from a .json file which holds information regarding the built model and also a database file that keeps a previous processed samples log.

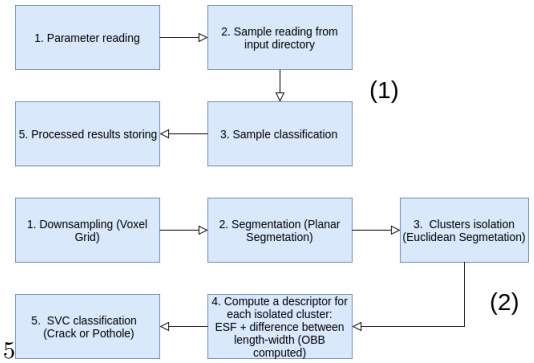

Fig. 6. (1) Classification application (2) Cropping pipeline.

Thus the general workflow starts reading input point cloud files (from a earlier setted up directory) and applying the sample cropping pipeline in order to get possible pavement depressions clusters. Then for each one build the custom descriptor and use the previously trained machine learning model, which is read from hard disk drive, to get the pavement depression type. As a result, every isolated cluster are stored in point cloud format in an output directory together with their computed width, length and depth. This process is pictured in Figure 6. In Figure 5.1 the general architecture is described, and in Figure 5.1 a screenshot of the application is shown.

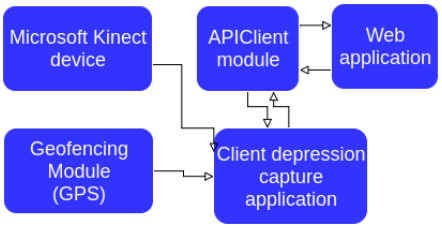

Fig. 7. General software modules from fault-capturing application.

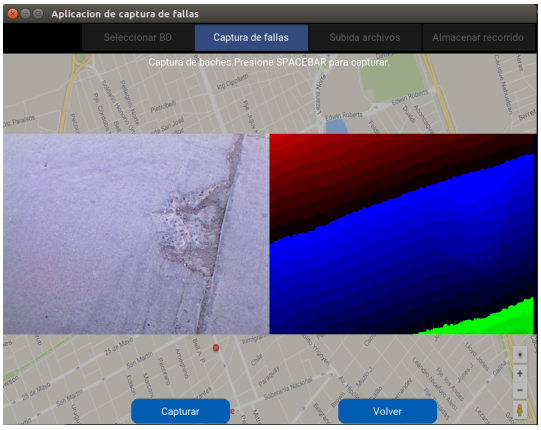

Fig. 8. Pavement depression measuring application interface (Left: RGB video view. Left: Nigth vision view.).

6 Conclusions and future work

In this work we have described a system for the collection and intelligent classification of potholes and cracks in pavement surfaces, using a low-cost commercial device along with the open-source PCL algorithms connected in an easily configurable pipeline. Moreover the estimated properties related to the pavement depression (width, length, depth), which may be useful for performing a material budget estimation in the repairing process, were successfully computed and stored for each isolated cluster. The system successfully classified collected data as potholes or cracks with a proper accuracy.

There exists other proposals on pavement depressions detection and analysis, which encompass researching and image processing with 3D videos and images, the following investigations have been made [2–6]. Alternatively, studies have been performed utilizing vibration devices such as accelerometers in [7–10]. Otherwise, studies in which the main focus is on specific road potholes features with aim of getting to process three-dimensional points coordinates as it is seen on [11, 12]. However none of them uses domestic devices nor classification algorithms based on machine learning. The use of automated learning is interesting since pavement faults tend to observe similar qualities within a city, as a consequence of climatic variations, used materials and quality of human work.

Future work has several directions. A continuous capture is desirable since it will make possible to avoid stopping the car. This requires a proper adaptation of dynamic gathering of data, since it demands real-time algorithms for point-cloud noise reduction. We are also interested in the estimation of materials and costs of repairing pavement faults. Since there is a 3D model of every fault a close,

reliable estimation can be produced. This can be done after processing a whole street or neighbourhood, hence helping to anticipate financial requirements and proper planning for e-governance. Even more, after long periods of time historic data can be mined to eventually detect areas that require special solutions other than normal patches to pavement. Finally, the addition of more than one type of sensor can be helpful to allow classification of faults which does not have enough depth to be fully captured by one sensing device.

References

1. Point Cloud Library (PCL), `https://www.pointclouds.org`
2. Korch y Brikalis: Pothole detection in asphalt pavement images. Advanced Engineering in Informatics (2011)
3. Buza, Omanovic, Huseinovic: Pothole detection with image and spectral clutering. Recent Adances in Computer Science and Networking (2013)
4. Seing-Ku Ryu , Taehyong Kim, Young-Ro Kim: Image-Based Pothole Detection System for ITS Service and Road Management System. Mathematical problems in Engineering (2015)
5. S Nieber , M Brooysen, R Kroon: Detection Potholes using Simple Processing Techniques and Real-World Footage. Proceedings of the 34th Southern African Transport Conference (2015)
6. Tomoyuki Yamaguchi , Shingo Nakamura, Ryo Saegusa, Shuji Hashimoto: Image-based crack detection for real concrete surfaces. Wiley InterScience (2008)
7. Eriksson, Girod, Hull, Newton, Madden, Balakrishnan: The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring. MobiSys (2008)
8. K. Chen, M. Lu, X. Fan: Road Condition monitoring using on-board three-axis acelerometer. 6th International ICST Conference on Communications and Networking in China (2011)
9. A. Tecimer, Z. Taysi, A. Yavuz, M. Karsligil Yavuz: Assessment of vehicular transportation quality via smartphones. Turkish Journal of Electrical Engineering and Computer Science (2013)
10. D. Casas-Avellaneda, J. López-Parra: Detection and localization of potholes in roadways using smartphones. DYNA Magazine (2016)
11. Sikai Xie: 3D pavement surface reconstruction and cracking recognition using Kinect-Based solution. The University of New Mexico (2015)
12. S Mathavan, M. Rahman, K. Kamal, S. Usman, I. Moazzam: Metrology and Visualization of Potholes using the Microsoft Kinect Sensor. 16th International IEEE Annual Conference on Intelligent Transportation Systems (2013)
13. H. Brink, J. W. Richards, M. Fetherolf: Real-World Machine Learning. Mannnig Publications (2017)
14. Scikit Learn (Support Vector Machines), `http://scikit-learn.org/stable/modules/svm.html`
15. Tom M. Mitchell: Machine Learning. McGraw-Hill (1997)