Computers and Electronics in Agriculture 154 (2018) 71–79

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

LiDAR-only based navigation algorithm for an autonomous agricultural robot

Flavio B.P. Malavazi, Remy Guyonneau*, Jean-Baptiste Fasquel, Sebastien Lagrange, Franck Mercier

LARIS, System Engineering Research Laboratory of Angers, University of Angers, France

ARTICLE INFO

Keywords:
Crop navigation
LiDAR measurements
Line extraction
PEARL
RANSAC

ABSTRACT

The purpose of the work presented in this paper is to develop a general and robust approach for autonomous robot navigation inside a crop using LiDAR (Light Detection And Ranging) data. To be as robust as possible, the robot navigation must not need any prior information about the crop (such as the size and width of the rows). The developed approach is based on line extractions from 2D point clouds using a PEARL based method. In this paper, additional filters and refinements of the PEARL algorithm are presented in the context of crop detection. A penalization of outliers, a model elimination step, a new model search and a geometric constraint are proposed to improve the crop detection. The approach has been tested over a simulator and compared with classical PEARL and RANSAC based approaches. It appears that adding those modification improved the crop detection and thus the robot navigation. Those results are presented and discussed in this paper. It can be noticed that even if this paper presents simulated results (to ease the comparison with other algorithms), the approach also has been successfully tested using an actual Oz weeding robot, developed by the French company Naio Technologies.

1. Introduction

The legislation about the use of chemical products for farming is getting increasingly severe. In France for instance, the Ecophyto 2018 program aims to drastically reduce the use of phytosanitary products (The ecophyto, 2018). As a result, some agricultural tasks that were ease (but still not easy) by the use of chemicals (weeding for instance) need alternative solutions to maintain the production efficiency. As a response to that need, the French company Naio Technologies[1] developed an autonomous weeding robot: the Oz robot. This robot is equipped with a LiDAR[2] sensor that is used to detect the crops, therefore allowing it to move autonomously inside the field without damaging the vegetables. For an efficient autonomous navigation, the robot (as provided by the company) needs some prior information about the length, the width and the number of the field crop rows. That is, the navigation behavior is directly dependent of the accuracy of those informations. The objective of the work presented here is to provide a new autonomous navigation algorithm that does not require any prior field information.

While expecting an autonomous robot navigation, the first results were obtained with camera based systems (Gerrish and Surbrook, Reid and Searcy, 1987). But as pointed out in Hiremath et al. (2014), the camera data are sensitive to light conditions and atmospheric effects, which can affect the robustness of the approach. An other approach is to consider a GPS[3]-based navigation (Bell, 2000; Pérez-Ruiz et al., 2012). But unless improved precision is considered, RTK[4]-GPS for instance, classical sensors are not accurate enough for navigation purposes. Moreover RTK-GPS can be expensive and are not adapted for an Oz robot size/price system.

LiDAR based approach appears to be an affordable alternative while being weakly sensitive to outdoor lighting, that is why it is considered in several commercial robots (the Oz robot, but also the new French robot PUMAgri[5] for instance). As mentioned before, in addition to the sensor data the current robots need some prior information about the

* Corresponding author.
E-mail addresses: flavio.barrospimentelmalavazi@etud.univ-angers.fr (F.B.P. Malavazi), remy.guyonneau@univ-angers.fr (R. Guyonneau), jean-baptiste.fasquel@univ-angers.fr (J.-B. Fasquel), sebastien.lagrange@univ-angers.fr (S. Lagrange), franck.mercier@univ-angers.fr (F. Mercier).

1 http://www.naio-technologies.com/en/.
2 Light Detection And Ranging.
3 Global Positioning System
4 Real-Time Kinematics.
5 http://www.sitia.fr/innovation-robotique/plateforme-pumagri/.

https://doi.org/10.1016/j.compag.2018.08.034
Received 19 February 2018; Received in revised form 22 August 2018; Accepted 26 August 2018

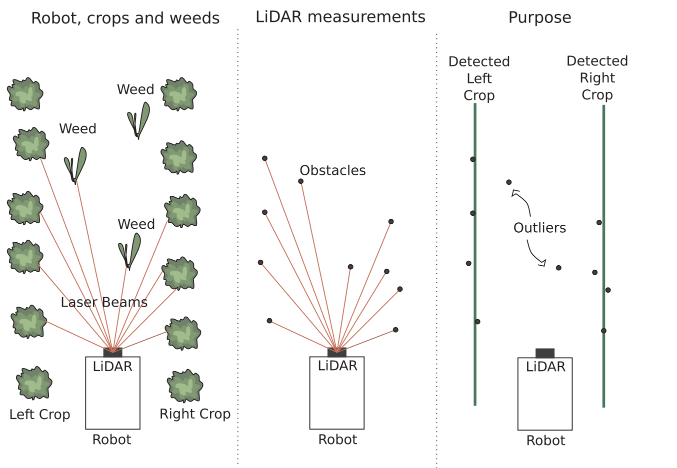

Fig. 1. The considered line fitting problem.

crops (size, length…) and are dependent to the accuracy of those informations. That is, this paper focuses on processing LiDAR data to propose a robust autonomous navigation method that does not need those prior information.

To move autonomously in the field, the robot must detect the crop rows. This can be cast into a problem of model fitting: from a data set (LiDAR measurements), we have to be able to find a set of straight lines (the rows) that best fit the data into individual clusters (Fig. 1).

In Hiremath et al. (2014), an interesting LiDAR based autonomous navigation algorithm is presented. The main drawback of this approach is that it requires a ”testing phase” in order to calibrate the algorithm. This can be assimilate to prior knowledge requirement, that we want to avoid for robustness purposes.

The considered approach in this paper is based on line detection (the crops) in a 2D point cloud (LiDAR measurements), as it can be done in Barawid et al. (2007). Two famous approaches for line detection are RANSAC-based line fitting (Fischler and Bolles, 1981) and Hough transform (Barawid et al., 2007; VC, 1962). From Jacobs et al. (2013) it appears that RANSAC-based approaches are generally more efficient than Hough transform to detect lines in a 2D point cloud.

The recently proposed PEARL Algorithm (Isack and Boykov, 2012) appears to be more efficient than RANSAC. Nevertheless, this algorithm depicts limitations. In this paper, we propose a refined PEARL algorithm that overcome the limitations of the initial PEARL algorithm, using a penalization of outliers, a model elimination step, a new model search and a geometric constraint. Furthermore, a navigation algorithm based on this refined PEARL is proposed.

The paper is organized as followed. Section 2 details the methods considered in this paper, starting with the original PEARL approach then presenting our refined PEARL algorithm and finally introducing the navigation algorithm based on it. Section 3 focuses on experiments performed using the Oz robot simulator, including a comparison with other navigation algorithms. Finally, Section 4 discusses about this work and results while Section 5 concludes this paper and presents perspectives.

2. Method

This section presents the original PEARL method as detailed in Isack and Boykov (2012), then our refined version of it to detect crops and finally a navigation algorithm based on this crop detection.

2.1. The original PEARL algorithm

2.1.1. The general idea of the approach

PEARL is an iterative approach (as RANSAC) that was found to be promising due to the fact that it usually converges in less iterations than RANSAC (Isack and Boykov, 2012). This is very interesting for real time applications such as the one at hand. It does so by using the knowledge of the last iteration when computing the new one while RANSAC starts over every time waiting for the residuals to be under a threshold.

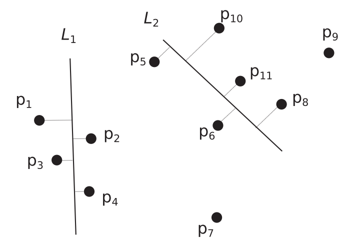

Fig. 2. Labels example: $\{p_1, p_2, p_3, p_4\}$ are labeled with the model L_1, $\{p_5, p_6, p_8, p_{10}, p_{11}\}$ are labeled with the model L_2 and $\{p_7, p_9\}$ are labeled with the model $L_\varnothing$. For instance $L(p_2) = L_1$ and $L(p_8) = L_2$.

PEARL method aims at minimizing a function, called *energy* (Eq. (1)). This function represents a *score* for a set of models (in our case a model corresponds to a line) according to a data set of points. In other words, it allows to compare two sets of models for the same data points and thus to select the one that best fit the data. The energy E is defined by

$$E(\mathscr{L}_i) = \sum_p ||p{-}L(p)|| + \lambda\cdot \sum_{(p,q)\in\mathscr{N}} w_{pq}\cdot\delta(L(p) \neq L(q)), \tag{1}$$

where

- $\mathscr{L}_i = \{L_j\}\bigcup L_\varnothing$ is the current set of models, L_j: $f_j(x) = a_j x + b_j$ is the j^{th} model (the j^{th} line) and $L_\varnothing$ the empty model. The empty model is used for points that are not associated to a line (thus considered as outliers, Fig. 1). Fig. 2 presents an illustration of points associated to models;
- $p \in \mathbb{R}^2$ is a point extracted from the LiDAR sensor data and $L(p) \in \mathscr{L}_i$ is the model associated to the point p (i.e. L: $p \to L_j \in \mathscr{L}_i$);
- $||p{-}L(p)||$ is the euclidean distance between the point and its associated line;
- $\mathscr{N}$ is the set of neighbor points and an element $(p, q) \in \mathscr{N}$ corresponds to two points p and q in the same neighborhood such that p is associated to the model $L(p)$ and q is associated to the model $L(q)$;
- $\delta(L(p) \neq L_q)$ equals 1 if $L(p)$ and $L(q)$ are not the same model, 0 otherwise;
- $\lambda\cdot\sum_{(p,q)\in\mathscr{N}} w_{pq}\cdot\delta(L(p) \neq L(q))$ is a penalty for the placement of close points in different models. This penalty is weighted using

$$w_{pq} = \exp\frac{-||p{-}q||^2}{\zeta^2}, \tag{2}$$

with $||p{-}q||$ being the euclidean distance between the points p and q;
- ζ and λ are two constants chosen heuristically (Isack and Boykov, 2012).

Note that outliers are points that are too far from any computed models according to an heuristically defined threshold. It corresponds to LiDAR points that are generated by an obstacle in the middle of the field (a weed for instance) and does not belong to any crop. That is, inliers are points associated with a model.

2.1.2. The algorithm steps

Here we present the original PEARL algorithm steps. This algorithm, detailed in Fig. 3, searches for models (lines in our case) in a data set (LiDAR points).

The detailed algorithm steps:

1. At initialization, the algorithm randomly samples data to get $\mathscr{L}_0$, which is the first set of models, according to a defined number of initial models. It may also add the empty model $L_\varnothing$ for points that