Optics and Lasers in Engineering 86 (2016) 98–105

Contents lists available at ScienceDirect

Optics and Lasers in Engineering

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

A MAP estimator based on geometric Brownian motion for sample distances of laser triangulation data

Markus Herrmann *, Marius Otesteanu

Politechnica University of Timisoara, Communications Department, Bd. Vasile Parvan, no. 2, 300223 Timisoara, Romania

ARTICLE INFO

Article history:
Received 1 July 2015
Received in revised form
9 May 2016
Accepted 17 May 2016
Available online 2 June 2016

Keywords:
Laser-stripe sensor
Optical triangulation
MAP
Maximum a posteriori estimation
Brownian motion
Geometric Brownian motion
Prediction
Line detection

ABSTRACT

The proposed algorithm is designed to enhance the line-detection stability in laser-stripe sensors. Despite their many features and capabilities, these sensors become unstable when measuring in dark or strongly-reflective environments. Ambiguous points within a camera image can appear on dark surfaces and be confused with noise when the laser-reflection intensity approaches noise level. Similar problems arise when strong reflections within the sensor image have intensities comparable to that of the laser. In these circumstances, it is difficult to determine the most probable point for the laser line. Hence, the proposed algorithm introduces a maximum a posteriori estimator, based on geometric Brownian motion, to provide a range estimate for the expected location of the reflected laser line.

1. Introduction

This paper introduces a novel method to enhance line detection in laser-stripe sensors when circumstances are ambiguous or uncertain. Laser-stripe sensors are capable of measuring a vertical section of a surface related to an internal-sensor coordinate system using a laser-line projection observed by an internal camera. Much research has been undertaken on laser-stripe sensors, with one of the first publications introducing the measurement technique in the mid-eighties [3]. Currently, applications for such sensors are ubiquitous, covering engineering fields such as industrial automation, geodetic measurements, computer vision (3D modelling), and robot guidance. When laser-stripe sensors are properly calibrated, i.e. the extrinsic parameter between camera and laser and the intrinsic parameter of the camera are well known, it is possible to transform pixel information of the laser-line reflection in the camera into real-world measurements within the sensor-coordinate space [13]. This is possible given that for every position within the camera image, there exists a unique ray intersecting the laser plane. This intersection point represents the real-world coordinate, which is computable if the camera calibration is known. To transform a coordinate from the image space to the sensor-coordinate space, a linear operation is applied and represented by a homography matrix [5,9]. However, since the images are slightly distorted due to the lens system, it is necessary to model lens distortion in order to first untwist the images.

Various calibration methods for laser-stripe sensors have been published [2,12].

Fig. 1 depicts a laser-stripe sensor observing a calibration body on the left-hand side and the associated measurement image on the right-hand side. Since the laser projection is observed by a camera, one fundamental task of such sensors is extraction of pixel coordinates from the laser-line reflection within the camera image. Unfortunately, limited research exists concerning line-detection algorithms. Currently, most commercial devices continue to use standard techniques introduced in the early nineties [6]. Small changes have been proposed, e.g. alternative methods for extracting the centre line of the laser [14]. Among the few works concerning image noise, Quing-Yang et al., introduced a method to remove environmental noise by subtracting two images [8]. There remains a lack of methods for enhancing the stability of laser-line detection on dark or reflective surfaces involving dynamic noise caused by the laser. Although strong laser reflections leave bright, easily extractable signals within the camera image, laser-line detection becomes challenging when additional reflections are visible within the image or, in the case of dark surfaces, if the

* Corresponding author.
E-mail addresses: info@mherrmann.org (M. Herrmann), marius.otesteanu@upt.com (M. Otesteanu).

http://dx.doi.org/10.1016/j.optlaseng.2016.05.009
0143-8166/

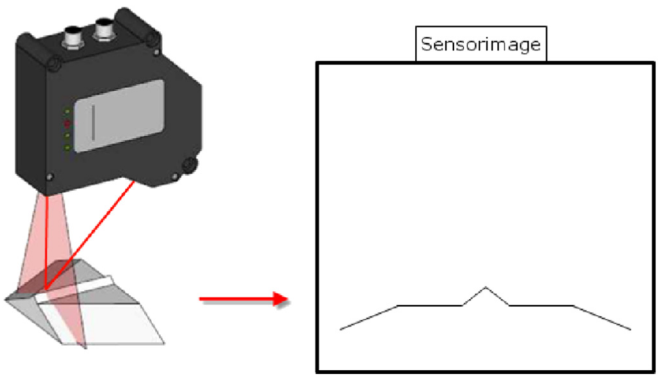

Fig. 1. Laser-stripe Sensor. The bright red region indicates the laser curtain and the dark red line is the empirical path of a single beam. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

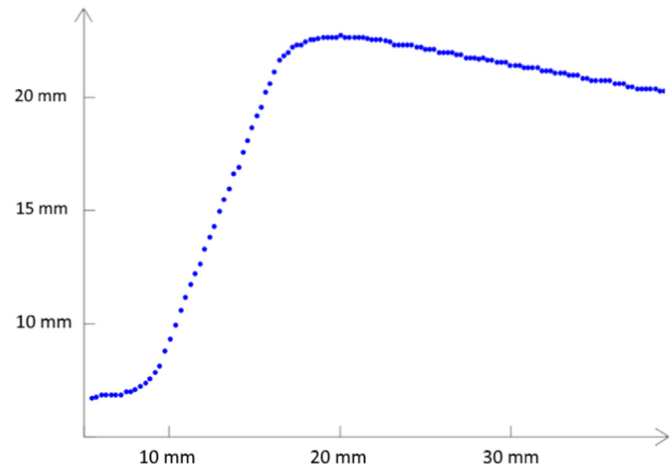

Fig. 2. A typical car-body scan showing differences in sample densities.

reflection intensity approaches noise level. Given the growing industrial popularity of laser-stripe sensors, applications involving complicated situations are becoming more common. Even in environments having strong reflective properties, the algorithm presented here can support line recognition in case of ambiguous situations involving contour gaps, jumps, or structural shadows. In such circumstances, an efficient method is required to determine the most probable candidate or decide whether a candidate exists. One common technique defines a threshold limiting the range for expected valid successors. If there are multiple candidates within the threshold, one strategy might be to consistently choose the neighbour candidate located closest to the latest laser reflection point. Unfortunately there is no equidistant spacing between samples and the closest point is not always the correct choice. In fact, the spacing between measured samples can vary due to the angular relation between the sensor and the object surface Fig. 2. One can expect the highest density of sample spots to exist where the sensor directly faces the object surface and lower sample density in regions where the angle between the surface and the sensor offsets from 90°. In order to address these situations, it is necessary to consider drift properties within the model equations. Here, we propose an algorithm based on geometric Brownian motion (GBM), an extension of Brownian motion [10] used in modelling asset-price behaviour in mathematical finance [10]. Given that sample distances share some similar properties with asset-price behaviour, it is possible to utilize GBM as a basis for sample-distance estimation. However, since line-detection logic is an integral part of the real-time software controlling the sensor hardware,[1] execution speed offers an additional challenge. This algorithm is intended to run on embedded systems in real-time environments, making avoidance of complex calculations or iterative root-finding algorithms a major requirement.

An additional application for the proposed algorithm involves scan segmentation. Given that points of discontinuities are detected automatically, the algorithm can be applied to determine separate segments within a scene. This can be useful for filtering, since points of discontinuities usually constitute critical spots, or for object detection or separation, since one segment usually represents one object within an observed scene.

2. Working principle

The algorithm is arranged into two logical steps, where the first step determines whether or not a measurement is within the estimated range for a successor. This step is not limited to the one successor, but can determine parameters for an arbitrary number of successors. Therefore, step one verifies whether only a few successors exceed estimated expected range by being classic outliers or if all upcoming successors are outside the expected range. In the event that there are many candidates for the next successor, the algorithm chooses the point with the highest probability, which is often the measure closest to the predicted mean. Points outside the limits defined by the predicted variance will be rejected as outliers.

The second step constitutes the update step, wherein a maximum a posteriori (MAP)-estimator is utilized to predict the most probable measurement state, $\bar{s}_i$. The prediction of $\bar{s}_i$ is based on its actual measurement, s_i, and its previous state, s_0. Therefore, $\bar{s}_i$ serves as a recursive input for subsequent prediction steps.

To determine a segmentation point, the algorithm determines if a well-defined number of measurements violates the expected sample distance. If so, it is probable that these points are not outliers, but rather define a new segment of contour points. In this case, the algorithm needs to be reinitialized with the first contour point of the new segment. In cases involving outliers, these points will be removed from the dataset. For model determination, it is necessary to assess the nature of the distances between samples. Given the simplest case, where the distances are constant and with normally-distributed noise superimposed, one can establish an initial model as:

$$p_s(s_i|\bar{s}_i, \sigma^2) = \left(\sigma \cdot \sqrt{2\pi}\right)^{-1} \exp\left(-\frac{(s_i - \bar{s}_i)^2}{2\sigma^2}\right) \quad (1)$$

where (1) s_i denotes spacing *(sample distance)*[2] at the ith measurement sample. The parameter, $\bar{s}_i$, denotes the expected spacing, predicted based on previous measures. Finally, σ^2 represents the variance between previous measures and their predictions. Thus, this initial model (1) represents a normal distribution, wherein both parameters represent the measurement expectation, $\bar{s}_i$, and the variance, σ^2, and are unknown and considered random variables. Therefore, the primary goal is to derive a model for these two parameters.

Starting with the spacing expectation, $\bar{s}_i$, it is possible to model it as its own random variable. Then, given a *probability density function* for $\bar{s}_i$ relative to the given data, s_i, it is possible to solve for $\bar{s}_i$, such that $p_{\bar{s}}(\bar{s}_i|\, s_i) \rightarrow \max$. Therefore, in order to solve for the most probable $\bar{s}_i$, the first step is to derive $p_{\hat{s}_i}(\bar{s}_i|\, s_i)$.

[1] Usually an embedded system with only limited resources.

[2] We use s (*spacing*) for the spacing between two samples instead of d, since d might be confused with the differential operator $\frac{d}{dx}$ later on.