UDC 004.031.2

ONLINE LANE DETECTION ALGORITHM FOR LINE SCAN CAMERA

Borys Tymchenko
O.M. Halchonkov, Ph. D.
Odessa National Polytechnic University, UKRAINE

ABSTRACT. At the introduced scientific article, we provide an algorithm for robust and stable for lane detection using only a single line scan camera. Our solution is directly applicable to the automatic car's concepts, when high-speed full-frame cameras are not useful either for their cost or for poor performance in road environment.

Automobile stability is critical while taking maneuvers at high speed or in dynamic environment. It affects both safety and comfort of driving. In this article, we propose theoretical algorithm and its implementation in model automotive car.

The aim of this work is to propose robust and fault-tolerant algorithm for lane detection using single-lined frame. It is suitable for applications, which need high-speed and low-resolution data about lane state (electronic stability control, lane departure control, etc.)

For implementation of the given algorithm, we used NXP Cup car model, which is a rear-wheeled car model at 1:12 scale. Line scan camera has 90 degrees field of view and installed at 26.5 cm height, that corresponds to 1.1 m height in the real-world scaled car. Car is riding track at 4.5 m/s that corresponds to 194 km/h speed of real-world scaled car.

The line scan camera module consists of a CMOS linear sensor array of 128 pixels and an adjustable lens. This camera has a 1x128 resolution. The lane track is a track consisted of white body (58 cm) and two black edges 2.5 cm each. Output of camera is linearly proportional to pixel brightness. Wide FOV lens combined with low camera resolution makes lane edges smaller than 1 px, so more sophisticated algorithms required [1].

Our algorithm has a pipeline structure, which consists of the following steps: image acquisition, noise reduction, edges enhancement, thresholding and dynamic lane extraction. Image acquisition is done in hardware, as a result we get a frame of raw pixel values, arriving at 1500 frames per second (FPS).

For noise reduction, we use mean filter with variable kernel size from 3 to 7, which depends on overall luminosity of the road. It allows us to get rid of surface inconsistency and shade edges, which can be mistakenly detected as lines.

For edge enhancement, we use custom developed kernel that combines 1st and 2nd derivatives of the input signal. Kernel is then convolved with previously denoised frame.

$$k = \{-2, -1, 7, -1, -2\} \qquad (1)$$

This kernel allowed us to come over subpixel line width to pixel size by magnifying small difference, which occurs on lane sides.

When we made edges visible, it is right time to reduce false positive results. To make it, we use special filter, introduced in [2]. Using disturbance curve on small windows (usually, ten probable line width), it is possible to detect sufficient outliers. With real road conditions, they can be sun glitches, oncoming traffic front lights and so on. In our implementation, we use 3σ rule for pixels: if pixel intensity lies more than 3σ from disturbance curve on window, it is considered the false positive.

As we have inconsistent background and various noise on the real track, these issues should be taken into account while developing algorithm and testing in idealized conditions. We used NICK adaptive thresholding [3], as it works as robust edge detector for inconsistent backgrounds and was originally used for OCR of ancient documents. Threshold value is computed with formula:

$$T = m + k\sqrt{\frac{\sum p_i^2 - m^2}{N}} \qquad (2)$$

Where k is a Niblack factor, m is a mean grey value for window, p_i is a pixel intensity and N is a window size. Niblack factor was chosen empirically and needs to be adjusted according to weather conditions. In our application, k is small negative number because we are interested in slightly lower threshold.

When we have our stable binary image that represents lane and side lines, we use some heuristics to achieve 95% failsafe detectors. This heuristic is represented with decision tree, which prunes false positive results using rules that describe standard track. We assume that between frames lines positions changed only a little, so we decided to sort line importance inversely proportional to its distance from line position at previous frame. In addition, we search for left and right edges of lane at the corresponding sides of detected lane center in previous frame.

Pruning algorithm involves cleanup of data that definitely cannot be a lane edge:

- Lane width is not in 75% confidence interval for lane width between the pair of found edges
- Both edges are on the same side from lane center at previous frame
- Edges are standard for given traffic code (width and color)

After pruning, we select edges that form a lane with center nearest to lane center at previous frame. These steps allowed us to differentiate lane in straight lines, crossroads, curves and hard corners with high robustness and reliability.

Conclusion. To test our algorithm, we took part in the championship. We took 1st place in semi-final of EMEA Cup. For about the year of testing, we faced almost none (< 1%) derailment through the fault of algorithm. As a result of this work, stable algorithm for online lane detection using single horizontal line was developed. Tests shown high robustness to weather conditions and perfect reliability with tolerance to false positives.

BIBLIOGRAPHY

1. Using Parallax TSL1401-DB Linescan Camera Module for line detection [Electronic source]. – Access mode: URL: http://cache.freescale.com/files/microcontrollers/doc/app_note/AN4244.pdf – Name from screen.

2. The Class of Generalized Hampel Filters [Electronic source]. – Access mode: URL: http://www.eurasip.org/Proceedings/Eusipco/Eusipco2015/papers/1570096433.pdf. – Name from screen.

3. Comparison of Niblack inspired Binarization methods for ancient document [Electronic source]. – Access mode: URL: https://pdfs.semanticscholar.org/76bd/5997bdd6d0a2611c590f5cf30f95c88e18ad.pdf – Name from screen.