On Detection of Yaw and Roll Angle Information for Vehicle Oblique Crash using Hough Transform

Sondre Sanden Tørdal, Andreas Klausen, Hamid Reza Karimi
Kjell G. Robbersmyr, Mladen Jecmenica and Ole Melteig

On Detection of Yaw and Roll Angle Information for Vehicle Oblique Crash using Hough Transform

Sondre Sanden Tørdal[1], Andreas Klausen[1], Hamid Reza Karimi[1]
Kjell G. Robbersmyr[1], Mladen Ječmenica[2] and Ole Melteig[2]

Abstract— **When performing vehicle crash tests, it is common to capture high frame rate video (HFR) to observe the vehicle motion during the impact. Such videos contain a lot of information, especially when it comes to geometric data. The yaw and roll angles from the HFR video is detected by using the Hough Transform and Matlab's Image processing Toolbox. The measured Yaw angle from the HFR video are compared with real life test data captured with a gyroscopic device inside the vehicle during the oblique vehicle impact.**

I. INTRODUCTION

The focus on safety when designing and building cars today are highly considered. And in order to test the safety for a vehicle impact crash, the traditional method is carried out by performing a physical crash test. From these tests it is common to capture high frame rate (HFR) video during the vehicle impact crash. These videos contain a lot of information, especially when it comes rotation of the vehicle during the impact. Such angles can be measured by means of rotational accelerometers placed inside the test vehicle. However these angles could also be detected from the HFR video by tracking line segments on the vehicle by using the Hough Transform (HT).

Today there are two main categories for modeling the vehicle for impact crash simulation; Lumped Parameter Modeling (LPM) and Finite Element Method (FEM). The first category by using the LPM may consist of up to several masses connected together by means of springs and dampers. This requires real life crash data to determine the parameters for both spring and damper. Jonsén et al. [1] used a lumped parameter model and real life crash data to identify the parameters to describe the crash. Otherwise it is also possible to identify the parameters by using computational power. Kim et al. [2] used an optimization algorithm to optimize and determine the LPM parameters. Also Pawlus et al. [3] used a pure analytic method to identify the parameters for a single mass, spring and damper system. Which is also known as the Kelvin model. The second category is using a FEM model to simulate the vehicle impact crash, however such simulations are computationally heavy and time consuming, but will in general give good and accurate results. Zaouk et al. [4] used a FEM model of a pick-up truck to simulate several impact crashes and compared the results with real life impact crash data.

Both these vehicle crash modelling categories requires to be validated. Such validation could be done by using the rotational angles obtained from the HFR movie. Fernandes et al. [5] has improved the HT algorithm, which in the first place is a popular tool for line detection due to its robustness to noise and missing data. The HT was also used to detect lines in real-time, and this states the efficiency of using the HT for line detection. Ji et al. [6] also states that straight lines detected by the HT are frequently adopted as a geometric element in high level image processing or object detection. Also Greenspan et al. [7] proved that the HT is a useful and robust tool to track object in a time sequence of images such as a video.

In this paper the global rotational angles from a HFR video captured during an oblique bus impact will be extracted using the HT. The aim of this paper is to obtain the rotational data for the bus during the impact, since this data will be useful to verify the impact models based on LPM and FEM. As well as the data can be used for model parameter identification in more advanced three-dimensional LPM models.

II. HOUGH TRANSFORM

The Hough Transform [8] is a statistical algorithm which is suitable for extracting features in images such as straight lines. This algorithm is a voting process which compares all possible lines in an image with $n \times m$ pixels. The algorithm will then return the indicated lines in the image, based on predefined thresholds set by the user. There are two main user defined thresholds which will determine which Hough lines that will be detected or not. The first threshold sets a lower boundary for the minimum length of the indicated Hough lines. The second threshold describes the upper boundary for the gap length between two detected lines, if this gap length is smaller than the boundary it will merge the two lines together. The algorithm will define the indicated lines in the image as follows:

$$\rho = x\cos\theta + y\sin\theta \tag{1}$$

Where (x, y) are the measured coordinates in the image, $\theta(-\pi/2 \leq \theta \leq \pi/2)$ is the angle between the normal axis and the x-axis, and ρ is the normal distance from the origin. Fig. 1 illustrates the geometric relationship between the indicated line segment and the reference coordinate system (Origin, O).

[1] S. Sanden Tørdal, A. Klausen, H.R Karimi, K.G Robbersmyr are with the department of Engineering of the faculty of Engineering and Science, the University of Agder, N-4898 Grimstad, Norway, e-mail: hamid.r.karimi@uia.no

[2] M. Ječmenica and O. Melteig are with the Institute for Process Technology of the Department of Technology, Telemark University College, N-3914 Porsgrunn, Norway

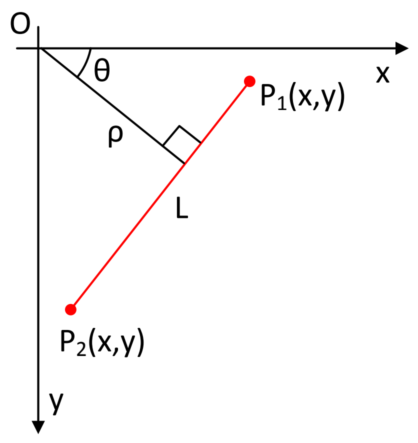

Fig. 1. Geometric relationship between the indicated line and origin.

Point $P_1(x, y)$ and $P_2(x, y)$ are also shown in Fig. 1. These points are used to specify the two endpoints of the indicated finite Hough line. This information is useful for calculating the angle between the indicated line segment L and the x-axis. However, Matlab includes predefined functions for the Hough transform which can be used to extract the angle of yaw and roll motions of the vehicle as a function of time.

Example: Edge Line Detection of a Square

The Hough transform can easily be explained by using a simple example where four lines should be detected at the edges of a square which is rotated $40°$ clockwise. The rotated square is shown in Fig. 2, both with and without the red indicated lines.

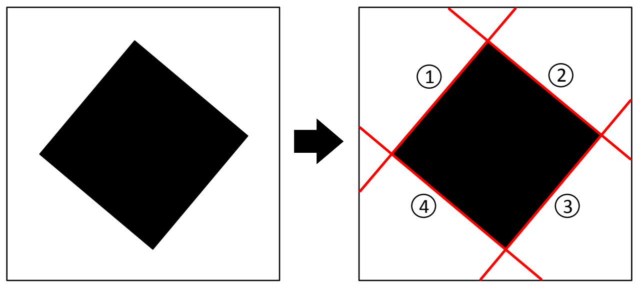

Fig. 2. Houg line detection of rotated square using the Hough transform.

The result of the Hough transform can be illustrated as a histogram containing the four indicated houghpeaks as shown in Fig. 3.

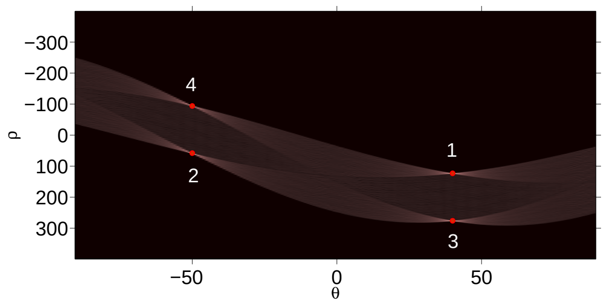

Fig. 3. Histogram of the Hough transform with indicated houghpeaks.

This four houghpeaks are represented with numbered red dots in the histogram, where the numbers represents the four lines indicated at the edges of the rotated square shown in Fig. 2. Each of the four peaks are located at given values for θ and ρ which corresponds to the coordinate system shown in Fig. 1

III. DETECTION METHOD

A. System Description

During the oblique bus collision, HFR videos were captured from three different positions; from the front, the back and the top. The camera from the back contains lower quality video than the camera from the front. Therefore the video captured from the top and front are used to detect the global angles of the bus during the oblique impact. A simple illustration of the test setup is shown in Fig. 4, where the bus, the two camera directions, the local and global coordinate systems are illustrated.

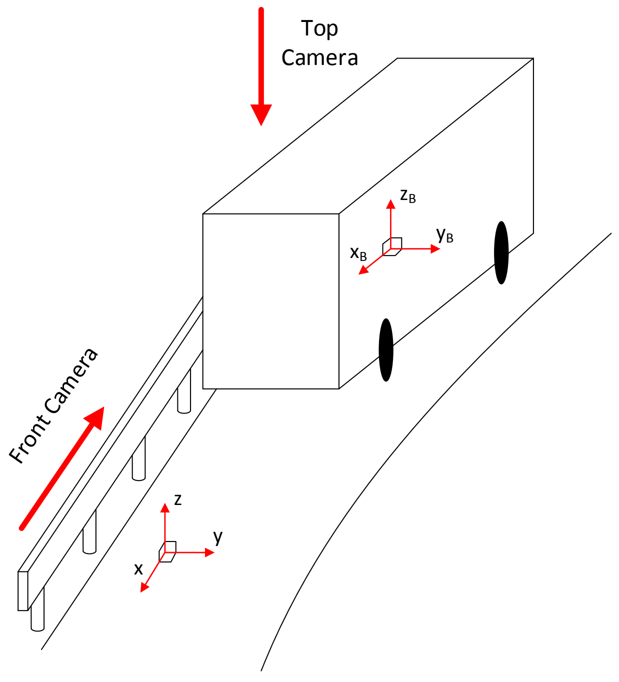

Fig. 4. Bus impact test set up with HFR camera directions.

Both videos are captured with a frame rate of 250 frames per second (FPS), which should ensure that the angles measured from the videos will be of high resolution.

B. Detection Implementation

The HFR video captured of the bus during the oblique impact should be processed with the use of Matlab's Image Processing Toolbox and the Hough Transform function. Before the angle detection by the Hough transformation can start, the video has to be converted to a preferable video format. The MPEG-4 format is chosen since this format is supported by Matlab's video import function *VideoReader*, as well as MPEG-4 is less space demanding than uncompressed formats.

It is preferable to convert the video to grayscale video before the movie processing. This can be done by using the *rgb2gray* function in Matlab. The result of the converting and import process in Matlab is shown in Fig. 5, where also the area of interest is indicated with a red rectangle.

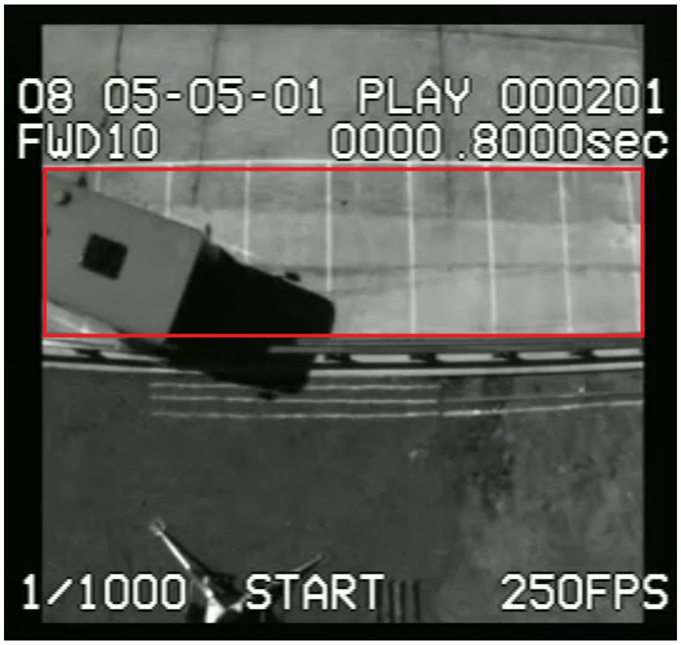

Fig. 5. 1st frame of original video.

By isolating a small region of interest, as shown in Fig. 5 the amount of image information is reduced significantly. This isolation is necessary to remove other straight lines which may have the same angle as the buss during the video. The resulting frame which is cropped to isolate the field of interest is shown in Fig. 6.

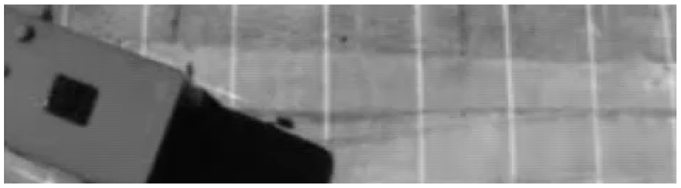

Fig. 6. 1st isolated frame of interest.

After that each single frame have been cropped and converted to grayscale, the frames are stored in Matlab for Hough line detection. It should also be mentioned that a Canny edge detector [9] has been used to improve the robustness of the Hough line detection. The result of this edge detection is not used when plotting the video frames together with the detected Hough lines.

IV. MATLAB ALGORITHM

When the HFR video is prepared and stored in Matlab as descirbed in Sec. III-B, the angles can be extracted from each single frame by implementing the HT. This detection procedure is done by establishing a Matlab algorithm which will process each single frame by using a single for loop. A flowchart describing the Matlab algorithm is shown in Fig. 7.

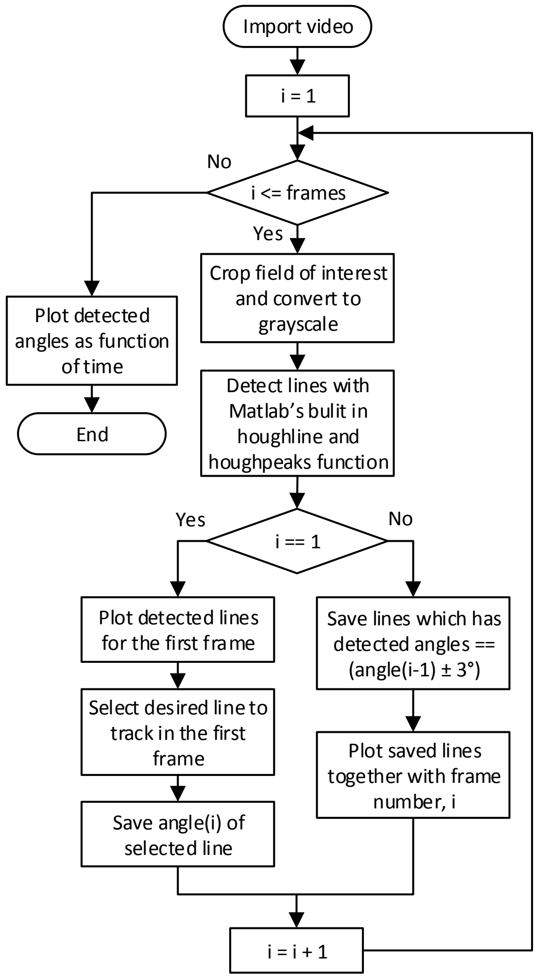
Import video
i = 1
i <= frames
No
Yes
Crop field of interest and convert to grayscale
Plot detected angles as function of time
End
Detect lines with Matlab's bulit in houghline and houghpeaks function
i == 1
Yes
No
Plot detected lines for the first frame
Select desired line to track in the first frame
Save angle(i) of selected line
Save lines which has detected angles == (angle(i-1) ± 3°)
Plot saved lines together with frame number, i
i = i + 1

Fig. 7. Flowchart of Matlab algorithm.

After that the first frame has been processed by the Matlab algorithm, the first frame together with the first set of indicated lines will be plotted together in a figure, and the algorithm pauses. This enables the user of the algorithm to select the line which aligns at the roof edge of the bus. The Matlab figure showing the selection of lines detected in the first frame is shown in Fig. 8.

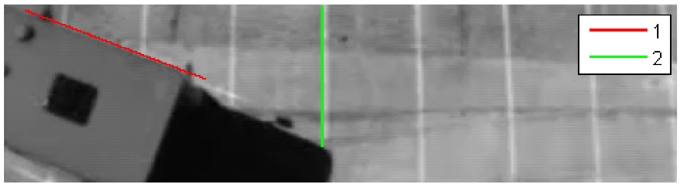

Fig. 8. 1st. video frame used to choose desired edge to track.

As shown in Fig. 8, it is reasonable to track the red line through the HFR video. This angle should then represent the yaw angle of the bus as a function of time.

When the user have selected the desired edge to track in the first frame, the algorithm stores the detected angle temporary. This temporary angle is used for filtering out lines in the next frame which is not within the range of the temporary angle plus a threshold set to $\pm 3°$. The temporary angle is then updated by the average angle of the lines within the threshold. This process is repeated for all the frames and the result of this algorithm is shown in Fig. 9 where some selected frames are shown during the processing.

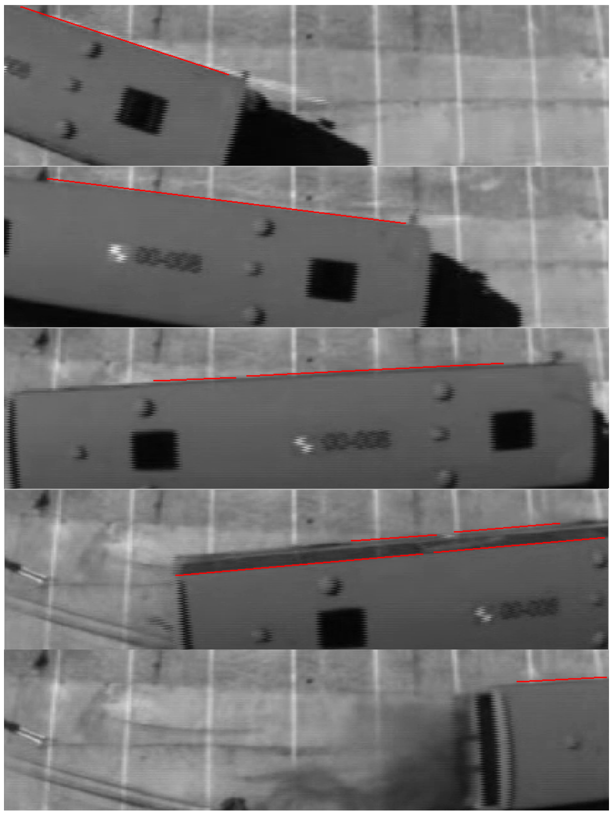

Fig. 9. Selected frames with detected lines through the HFR video.

During the processing, the angle detected from each frame is saved and can therefore be plotted as a function of time. These angles are also numerically differentiated to obtain information for angular velocity and acceleration.

V. EXPERIMENTAL RESULTS

The result of implementing the HT from Sec. II in the Matlab algorithm described in Sec. IV is that the yaw angle as well as the global angle from the front can be obtained as function of time. To ensure that the obtained angles from the HFR video is correct they should be compared to other measurements of the angles. From the real life oblique bus impact test, Robbersmyr et al. [10] was logging the yaw rate in deg/s for the bus by using a gyroscopic device placed in the center of gravity inside the bus. This measured data can be used for comparing the measured data from the HFR video and measured angular velocity. By numerical integration of the angular velocity, the angle of the bus can be obtained.

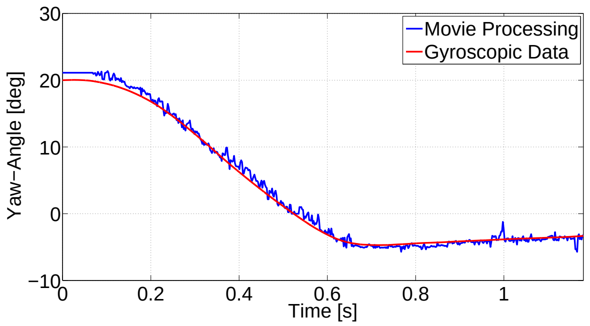

Fig. 10. Comparison of yaw rotational angle vs. gyroscopic data.

In Fig. 10 the measured angle from the HFR video is compared with the numerically integrated angle obtained from the gyroscopic data. The resulting curves are quite similar, but some deviation can be observed at the initial angles. To obtain data for the angular velocity from the HFR video, a polynomial was fitted to the discrete angular data. This polynomial was then differentiated to describe the angular velocity measured from the video.

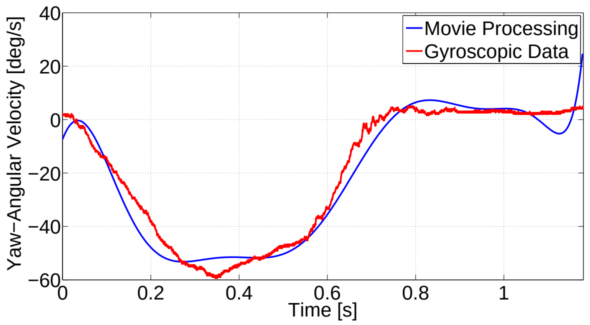

Fig. 11. Comparison of yaw rotational velocity vs. gyroscopic data.

When comparing the two angular velocities in Fig. 11, there are some more deviation between the two curves, but still the results are promising. The cause of inaccuracy may come from the simple method used for the differentiation of the yaw angle measured from the video. By applying a more accurate differentiation together with interpolation may lead to more accurate results.

The video was also used for detection of the global roll motion of the bus. This angle was extracted from the video captured in front of the bus during the impact. The camera location is illustrated in Fig. 4 and is indicated as the *Front Camera* on the illustration. The result of using the Matlab algorithm described in Sec. IV to extract the global roll motion of the bus is illustrated in Fig. 12.

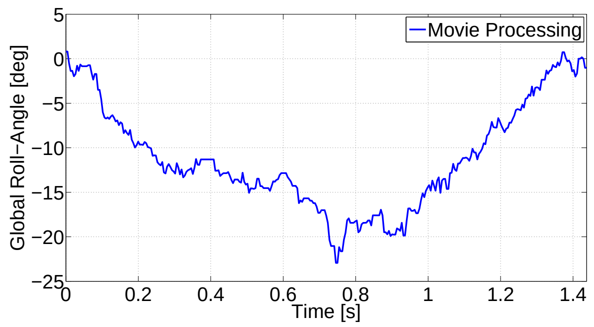

Fig. 12. Global roll angle from HFR video.

For this data there are no data set for comparison, since such measurements was not obtained from the real life crash test. However, when executing the Matlab algorithm the detected edges at the bus can easily be monitored by plotting the indicated edge in each movie frame. This will give a visual feedback to verify that the correct edge is tracked throughout the video.

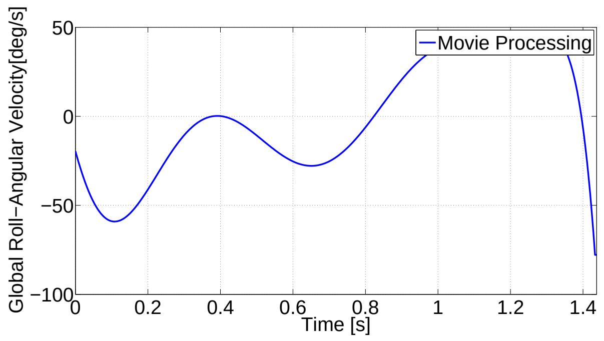

Fig. 13. Global roll velocity from HFR video.

In Fig. 13 the global roll angular velocity is plotted. This angle is also obtained by fitting a polynomial to the global roll angle data. This polynomial is then differentiated to describe the angular velocity of the roll motion.

VI. CONCLUSIONS

In this paper, the Hough Transform is introduced to detect yaw and roll angle information for an oblique crash. Specifically, high frame rate (HFR) videos are used to observe the vehicle motion during the impact. It was shown that the yaw and roll angles from the HFR video are detected by using the Hough Transform and Matlab's Image processing Toolbox. The measured Yaw angle from the HFR video were compared with real life test data captured with a gyroscopic device inside the vehicle during the oblique vehicle impact. As a further work, the information obtained from this research will be used to construct a 3D mathematical model for an oblique collision.

ACKNOWLEDGMENT

This work is supported by the grant of the collaborative research project between the University of Agder and Telemark University College.

REFERENCES

[1] P. Jonsén, E. Isaksson, K. G. Sundin, and M. Oldenburg. Identification of lumped parameter automotive crash models for bumper system development. *International Journal of Crashworthiness*, 14:6:533–541, 2009.

[2] C. H. Kim, A. R. Mijar, and J. S. Arora. Development of simplified models for design and optimization of automotive structures for crashworthiness. *Structural and Multidisciplinary Optimization*, 22-4:307–321, 2001.

[3] W. Pawlus, J. E. Nielsen, H. R. Karimi, and K. G. Robbersmyr. Mathematical modeling and analysis of a vehicle crash. *Proceedings of the 4th European Computing Conference*, pages 194–199, 2010.

[4] A. K. Zaouk, N. E. Bedewi, C. Kan, and D. Marzougui. Validation of a non-linear finite element vehicle model using multiple imoact data. 1996.

[5] L. A. F. Fernandes and M. M. Oliveira. Real-time line detection through an improved hough transform voting scheme. *Pattern Recognition*, 41:299–314, 2008.

[6] J. Ji, G. Chen, and L. Sun. A novel hough transform method for line detection by enhancing accumulator array. *Pattern Recognition Letters*, 32(11):1503 – 1510, 2011.

[7] M. Greenspan, L. Sahng, and P. Jasiobedzki. Efficient tracking with the bounded hough transform. *Computer Vision and Pattern Recognition*, 1:520–527, June 2004.

[8] P. V. C. Hough. Method and means for recognizing complex patterns, 1960.

[9] L. Ding and A. Goshtasby. On the canny edge detector. *Pattern Recognition*, 34:721–725, March 2001.

[10] K. G. Robbersmyr and O. K. Bakken. Impact test of safety barrier, test tb51 (reference no. 00-008). Technical Report 008, Agder Research, 2001.