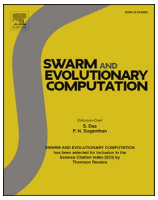
Swarm and Evolutionary Computation 41 (2018) 111–119

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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



Vanishing point detection for self-driving car using harmony search algorithm

Yoon Young Moon [a], Zong Woo Geem [b,*], Gi-Tae Han [a]

[a] *Dept. of Computer Engineering, Gachon University, Seongnam, South Korea*
[b] *Dept. of Energy IT, Gachon University, Seongnam, South Korea*

ARTICLE INFO

Keywords:
Self-driving car
Autonomous car
Lane detection
Vanishing point
RANSAC
Harmony search

ABSTRACT

Self-driving or autonomous vehicles require an ability to detect road lanes. In order to do so, a vanishing point should be first detected because the vanishing point exists on the extended lines of road lanes. For detecting the vanishing point, a random sample consensus (RANSAC) algorithm has been generally utilized. However, the performance of RANSAC is sometimes not so good but fluctuated. Thus, this study proposes a new approach to estimate the vanishing point using a harmony search (HS) algorithm. Results show that HS stably estimates vanishing points with respect to statistics when compared with RANSAC. We hope this model to be utilized in self-driving car in the future.

1. Introduction

A vanishing point is an intersection point at which all parallel lines in three-dimensional (3D) space meet together [1,13]. The vanishing point plays an important role in detecting road lanes because road lanes are parallel in real-world 3D space but they meet at the vanishing point in two-dimensional camera image [2]. Thus, the detection technology of vanishing point is necessary for self-driving automobiles.

In order to detect road lanes, all the edge components are first extracted from a camera image. Then, the vanishing point, which is the intersection point of the projected edge components, is estimated. Finally, based on the vanishing point, road lanes are determined [3].

So far various researchers have tackled the topic of vanishing point detection. Mainly, the research has focused on how accurately vanishing points could be detected when various noises exist [4–8,12,14]. Among them, the random sample consensus (RANSAC) algorithm, which utilizes straight lines obtained using Hough transform is one of the most popular ones [9,10]. The RANSAC algorithm decides the optimum model parameter by estimating noised data. It randomly selects any partial data from original data, and then identifies good model parameters after iterating the selection process many times [18,26–30].

RANSAC has the advantage of faster calculation speed [15–18,30]. However, because it employs random selection technique, it could have errors that create problems in real-life systems where accuracy matters. Thus, the objective of this study is to propose a more accurate vanishing point detection method by using the harmony search (HS) algorithm, which has been successfully applied to various problems so far [19–25].

This paper has the following structure: Section 2 introduces the RANSAC algorithm and vanishing point detection technique using it; Section 3 describes the proposed HS method and the procedure to detect the vanishing point by the method; Section 4 presents the results and analysis through the comparison between the RANSAC method and the HS method; and lastly Section 5 gives final conclusions.

2. RANSAC algorithm for vanishing point detection

RANSAC is a method that predicts the model from certain data sample [11]. RANSAC estimates the model parameters by finding the best hypothesis h_B among the set of all possible hypotheses **H** generated by the complete source data **U**. Such source data are typically contaminated by noise. In order to build the hypothesis h_i, a sample S_i is used.

RANSAC algorithm has the following structure:

Step 1. Construct a sample $S_i \subset \mathbf{U}$ consisting of s different elements.
Step 2. Build the hypothesis h_i based on the sample S_i.
Step 3. Evaluate the degree of agreement A_i of the hypothesis with the set of all source data **U**.

After the construction and evaluation of all **H** hypotheses, the hypothesis h_B with the best degree of agreement is chosen among them. The

[*] Corresponding author.
E-mail addresses: nunnengi@me.com (Y.Y. Moon), geem@gachon.ac.kr (Z.W. Geem), gthan@gachon.ac.kr (G.-T. Han).

https://doi.org/10.1016/j.swevo.2018.02.007
Received 31 January 2017; Received in revised form 26 January 2018; Accepted 11 February 2018
Available online 13 February 2018

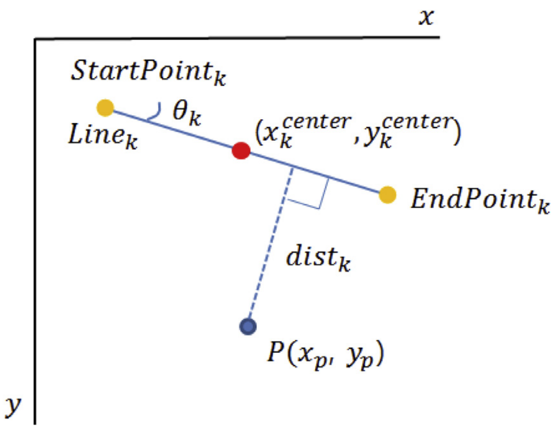


Fig. 1. Parameters for orthogonal distance between point and line.

maximization of the degree of agreement (number of inliers: A_i) is equivalent to the minimization of the penalty function whose value depends on the number of outliers. Therefore, the degree of agreement $A_i(\mathbf{U}, h_i)$ is computed as follows:

$$A_i = \sum_{k=1}^{K} Consensus_k \tag{1}$$

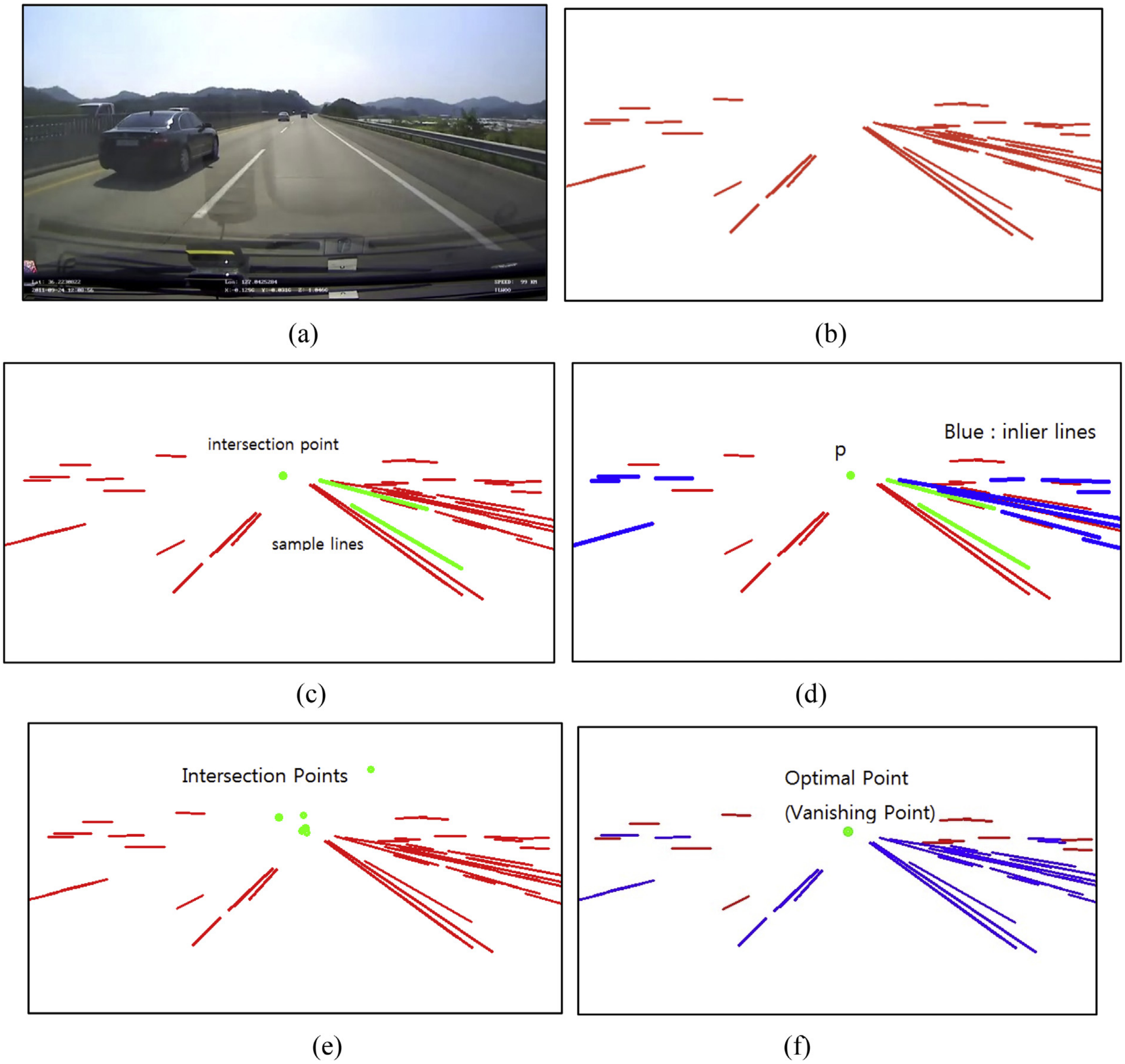
$$\text{when } Consensus_k = \begin{cases} 0\ (\theta > threshold) \\ 1\ (\theta < threshold) \end{cases},\ k \in \left[1, K\right],\ \theta$$

$: inlier\ measure\ element$

For vanishing point detection problem, the RANSAC algorithm tries to find a good model parameter (vanishing point) by randomly selecting some partial samples (parallel lines) that are constructed by Hough transform technique of road image [34]. Once a candidate vanishing point (intersection point of extended parallel lines) is chosen from random sampling, it is further tested whether it is inlier or outlier using the following equation:

$$dist_k = \frac{\left|\left(x_p - x_k^{center}\right)*\sin\theta_k - \left(y_p - y_k^{center}\right)*\cos\theta_k\right|}{\sqrt{\left(\cos\theta_k\right)^2 + \left(\sin\theta_k\right)^2}} \tag{2}$$

Using Equation (2), the distance between the candidate vanishing point and all parallel lines in a road image can be calculated. Fig. 1 [33] shows the distance-related parameters graphically.



(a) (b) (c) (d) (e) (f)

Fig. 2. Process of vanishing point detection by RANSAC method.

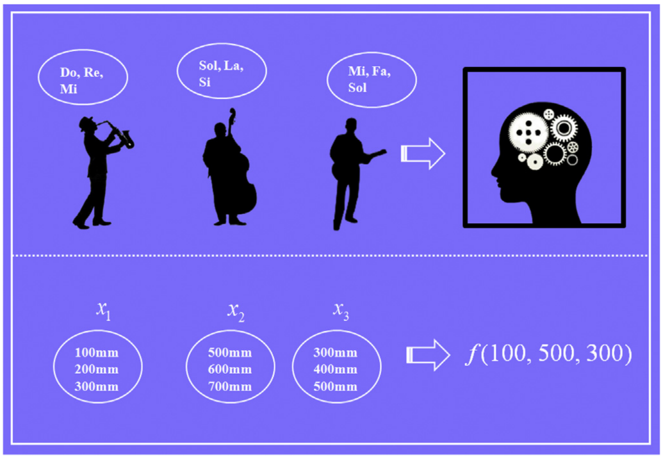


Fig. 3. Relation between musical jam session and numerical optimization.

The orthogonal distance between the straight line and any arbitrary point P can be used for deciding whether the point is inlier or outlier. If the distance is shorter than certain threshold, it can be classified as inlier. And, the number of inliers from all the parallel lines of the road image can be used to evaluate the quality of the candidate vanishing point as follows:

$$Count_p = \sum_{1}^{K} Consensus_k \tag{3}$$

when $Consensus_k = \begin{cases} 0 \ (dist_k > threshold) \\ 1 \ (dist_k < threshold) \end{cases}, \ k \in \left[1, K\right]$

The process of the vanishing point detection using RANSAC method is shown in Fig. 2 [9,10]: Fig. 2-(a) shows the original road image; Fig. 2-(b) shows result image of detected lines using Hough transform; Fig. 2-(c) shows an intersection point p (green) computed by two randomly selected straight lines (green); Fig. 2-(d) shows inlier lines (blue and green) and outlier lines (red) for the intersection point p; Fig. 2-(e) shows total intersection points after executing N (N = 32) times randomly (Here, if we want to make sure that at least one of random samples (sample size = two lines; and probability to select inliers = 50%) is an error-free set with probability 99.99%, we should make at least 32 samples [18]); and Fig. 2-(f) shows the optimal point selected after counting the number of inliers for each point.

3. Harmony search algorithm for vanishing point detection

In recent years, evolutionary algorithms that imitate a natural phenomenon or behavioral phenomenon have become prominent. The HS algorithm, which mimics music improvisation phenomenon, has been applied to various problems including image processing [23]. The research in Ref. [21] applied HS to cluster points to distinguish a brain region within MRI medical images. Fourie et al. [22] set up a window on objects within images, and then traced the objects using the Bhattacharyya distance coefficient and HS that looks for a histogram within the image of the window.

The principle of HS could be explained as in Fig. 3, where musical improvisation (i.e., jam session) and numerical optimization can be compared each other. As the music harmony is enhanced practice by practice, the solution vector is enhanced iteration after iteration. More detailed process of the HS algorithm [24] is as follows:

3.1. Parameter initialization

HS has the algorithm parameters such as harmony memory size (number of solution vectors in harmony memory (HM)), harmony memory consideration rate (HMCR), pitch adjusting rate (PAR), distance bandwidth (BW), and number of improvisations (NI; total number of iterations). It is obvious that an adequate selection for HS parameters would enhance the algorithm's ability to search for the global optimum under a high convergence rate.

3.2. Harmony memory initialization

Let $\mathbf{p}_i = \{p_i(1), p_i(2), \ldots, p_i(n)\}$ represent the i-th randomly generated harmony vector: $p_i(j) = l(j) + (u(j) - l(j)) \times rand(0,1)$ for $j = 1, 2, \ldots, n$ and $i = 1, 2, \ldots,$ HMS, where rand(0,1) is a uniform random number between 0 and 1. Then, the HM matrix is filled with the HMS harmony vectors as follows:

$$\mathbf{HM} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_{HMS} \end{bmatrix} \tag{4}$$

3.3. Improvisation of new harmony vector

A new harmony vector $\mathbf{p}_{New}$ is built by applying the following three operators: memory consideration, random selection, and pitch adjustment. Generating a new harmony is called as “improvisation” in HS. In the memory consideration step, the value of the first decision variable $p_{New}(1)$ for the new vector is chosen randomly from any of the values already existed in the current HM, that is, from the set $\{p_1(1), p_2(1), \ldots, p_{HMS}(1)\}$. For this operation, a uniform random number r_1 is generated within the range [0, 1]. If r_1 is less than HMCR, the decision variable $p_{New}(1)$ is generated through memory considerations; otherwise, $p_{New}(1)$ is obtained from a random selection between the search bounds $[l(1),\ u(1)]$. Once $p_{New}(1)$ is obtained from memory consideration operation, it can be further modified using pitch adjustment operation where the value of $p_{New}(1)$ is slightly changed. Values of the other decision variables, $p_{New}(2)$ to $p_{New}(n)$, are also chosen accordingly.

3.4. Harmony memory update

After harmony vector $\mathbf{p}_{New}$ is generated, the harmony memory is updated by the survival of the fit competition between $\mathbf{p}_{New}$ and the worst harmony vector $\mathbf{p}_W$ in the HM. Therefore, $\mathbf{p}_{New}$ will replace $\mathbf{p}_W$ and become a new member of the HM in case the fitness value of $\mathbf{p}_{New}$ is better than the fitness value of $\mathbf{p}_W$.

3.5. Termination

If the number of iterations reaches NI, the computation is terminated. Otherwise, it goes to 3.3.

The computational procedure of the above basic HS can be summarized as follows:

```
Step 1. Set the parameters HMS, HMCR, PAR, BW and NI.
Step 2. Initialize the HM and calculate the objective function value of each harmony vector.
Step 3. Improvise a new harmony $\mathbf{p}_{New}$ as follows:
  for ($j = 1$ to $n$) do
    if ($r_1 <$ HMCR) then
      Select randomly a number $a$ where $a \in \{1, 2, \ldots,$ HMS$\}$
      $p_{New}(j) \leftarrow p_a(j)$
      if ($r_2 <$ PAR) then
        $p_{New}(j) \leftarrow p_{New}(j) +$ BW$\cdot(2r_3 - 1)$ where $r_1, r_2, r_3 \in$ rand(0, 1)
      end if
      if $p_{New}(j) < l(j)$
        $p_{New}(j) \leftarrow l(j)$
      end if
      if $p_{New}(j) > u(j)$
        $p_{New}(j) \leftarrow u(j)$
```

(*continued on next page*)

(continued)

```
        end if
    else
        $p_{New}(j) = l(j) + r_4 \cdot (u(j) - l(j))$, where $r_4 \in$ rand(0, 1)
    end if
    end for
Step 4. Update the HM as $\mathbf{p}_W \leftarrow \mathbf{p}_{New}$ if $f(\mathbf{p}_{New})$ is better than $f(\mathbf{p}_W)$
Step 5. If NI is completed, the best harmony vector $\mathbf{p}_B$ according to its fitness value in
the HM is returned; otherwise go back to Step 3.
```

The flow chart of the proposed vinishing point detection procedure including harmony search is as in Fig. 4. In this study, edges are extracted from the input images, and then candidate vanishing points are estimated using straight line components. Out of many candidate points, the best point is selected as vanishing point using the HS algorithm.

To find the edges from images, Canny edge detector is used as in Fig. 5 [1,16], and to find straight lines from detected edges, Hough transform is used as in Fig. 6 [1,16].

For the vanishing point detection, the HS algorithm uses the following algorithm parameters:

1) HMS: Number of solution vectors. In the proposed method, it is set to 5 based on the results of experiments.
2) HMCR: Ratio between existing information inside HM and random information from full range. In this study, HMCR ratio is set to be more than 0.8 because the objective function shows a tendency to be unimodal.
3) PAR: Rate to further change the value obtained from memory consideration. In this study, it is set to 0.3.

As in Fig. 4, the procedure of vanishing point estimation by HS is as follows:

Step 1: Set algorithm parameter values (HMS, HMCR, and PAR) and fill random solution vectors (coordinates of candidate vanishing points) in the HM as many as HMS.
Step 2: Generate a new harmony (NH) by considering HMCR and PAR.
Step 3: Compare the objective function values of NH and worst solution in existing HM, then keep the better one in HM.
Step 4: If the condition for calculation termination is not satisfied, return to Step 2.
Step 5: Select the point that received the most agreements from the extended straight lines as the vanishing point.
Step 6: Go to Step 1 for detecting the vanishing point of the next frame. Instead of using random HM, the next frame uses current HM as initial HM.

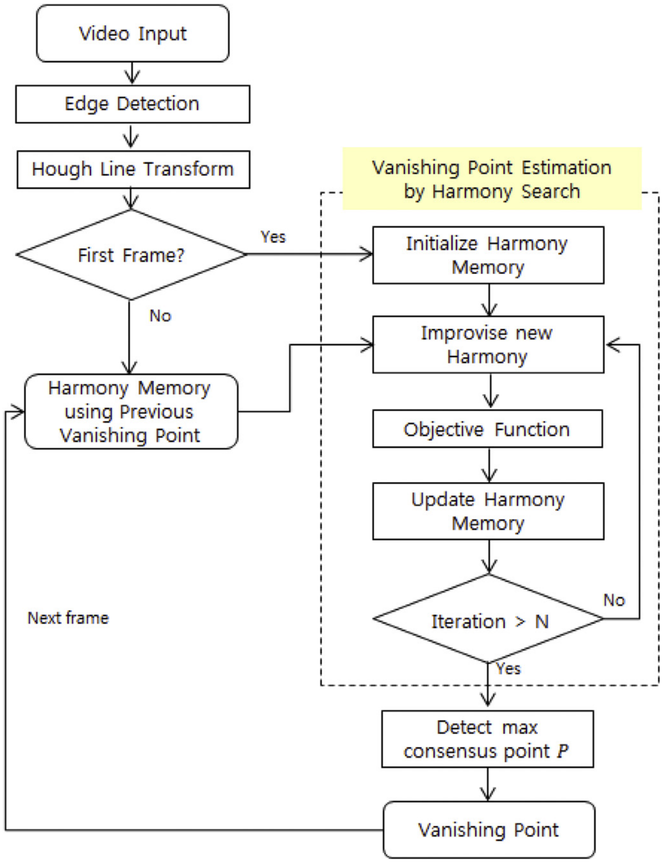


Fig. 4. Flow chart of harmony search-based vanishing point detection.

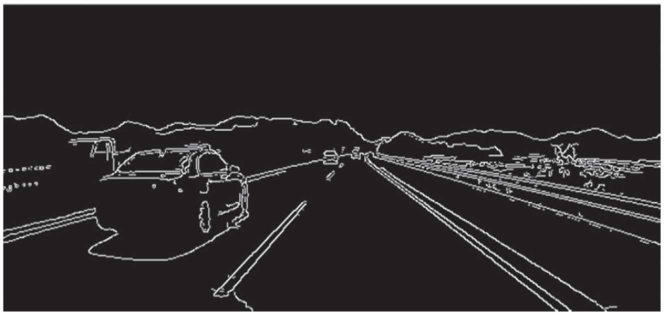


Fig. 5. Result of Canny edge detector process.

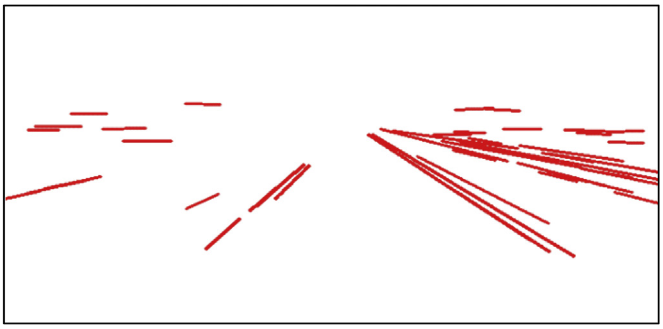


Fig. 6. Result of line detection by Hough transform.

```
(1) Configuration
        Set the parameters HMS, HMCR, PAR, BW, and NI.
(2) Initial Population
    (a) Build the harmony memory (HM) = $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{HMS}\}$
        where each individual $\mathbf{p}_i$ consists of coordinates of vanishing points.
    (b) Compute the fitness value $C_i$ for each $\mathbf{p}_i$.
        $C_i$ considers the whole available data U. Such fitness value is calculated by using
        a new objective function defined as:

$C_i = \sum_{1}^{K} Consensus_k$

$Consensus_k = \begin{cases} 0\ (dist_k > threshold) \\ 1\ (dist_k < threshold) \end{cases},\ k \in \left[1, K\right]$

where $dist_k$ is distance of $line_k$ and $P_i$
(3) Iterations $k = 1, \cdots,$ NI
    (a) Generate a new harmony $\mathbf{p}_{New}$(candidate solution) as follows:
            for ($j = 1$ to $n$) do
                if ($r_1 <$ HMCR) then
                    Select randomly an integer number $a$ where $a \in \{1, 2, \ldots,$ HMS$\}$
                    $p_{New}(j) \leftarrow p_a(j)$
                    if ($r_2 <$ PAR) then
```

(continued on next page)

(*continued*)

```
                $p_{New}(j) \leftarrow p_{New}(j) + \mathrm{BW}\cdot(2r_3-1)$ where $r_1, r_2, r_3 \in \mathrm{rand}(0, 1)$
            end if
            if $p_{New}(j) < l(j)$
                $p_{New}(j) \leftarrow l(j)$
            end if
            if $p_{New}(j) > u(j)$
                $p_{New}(j) \leftarrow u(j)$
            end if
        else
            $p_{New}(j) = l(j) + r_4\cdot(u(j)-l(j))$, where $r_4 \in \mathrm{rand}(0, 1)$
        end if
    end for
  (b) Calculate the fitness value ($\mathbf{p}_{New}$) which is total consensus number $C_{New}$.
  (c) Update the HM as $\mathbf{p}_W \leftarrow \mathbf{p}_{New}$ if ($\mathbf{p}_{New}$) is better than ($\mathbf{p}_W$).
(4) Optimal Solution
    The best coordinates $\mathbf{p}_B$, which has the highest consensus number $C_B$ in HM, is
    returned as optimal vanishing point.
```

This HS algorithm for vanishing point detection is mostly similar to the original HS algorithm [24]. However, this HS algorithm recurrently utilizes (recycles) final HM from previous frame as initial HM for current frame. In addition, while objective functions in many optimization problems are cost functions to be minimized, this problem utilizes degree of agreement as objective function.

An example of vanishing point detection using HS is graphically shown in Fig. 7: Fig. 7-(a) shows the initial HM. In this stage, HS chooses five arbitrary points; Fig. 7-(b) shows the worst harmony in HM (blue point from two blue straight lines); Fig. 7-(c) shows a new harmony made by HS (green point and blue straight lines); Fig. 7-(d) shows the updated HM after replacing the worst point with new point; Fig. 7-(e) shows the result of HM after 50 iterations; and Fig. 7-(f) shows final result (inlier lines are blue and outlier lines are red).

4. Computational results

The experiments presented in this study were conducted on an Intel i5 Dual Core CPU 1.7 GHz with 4 GB RAM. The resolution of the video

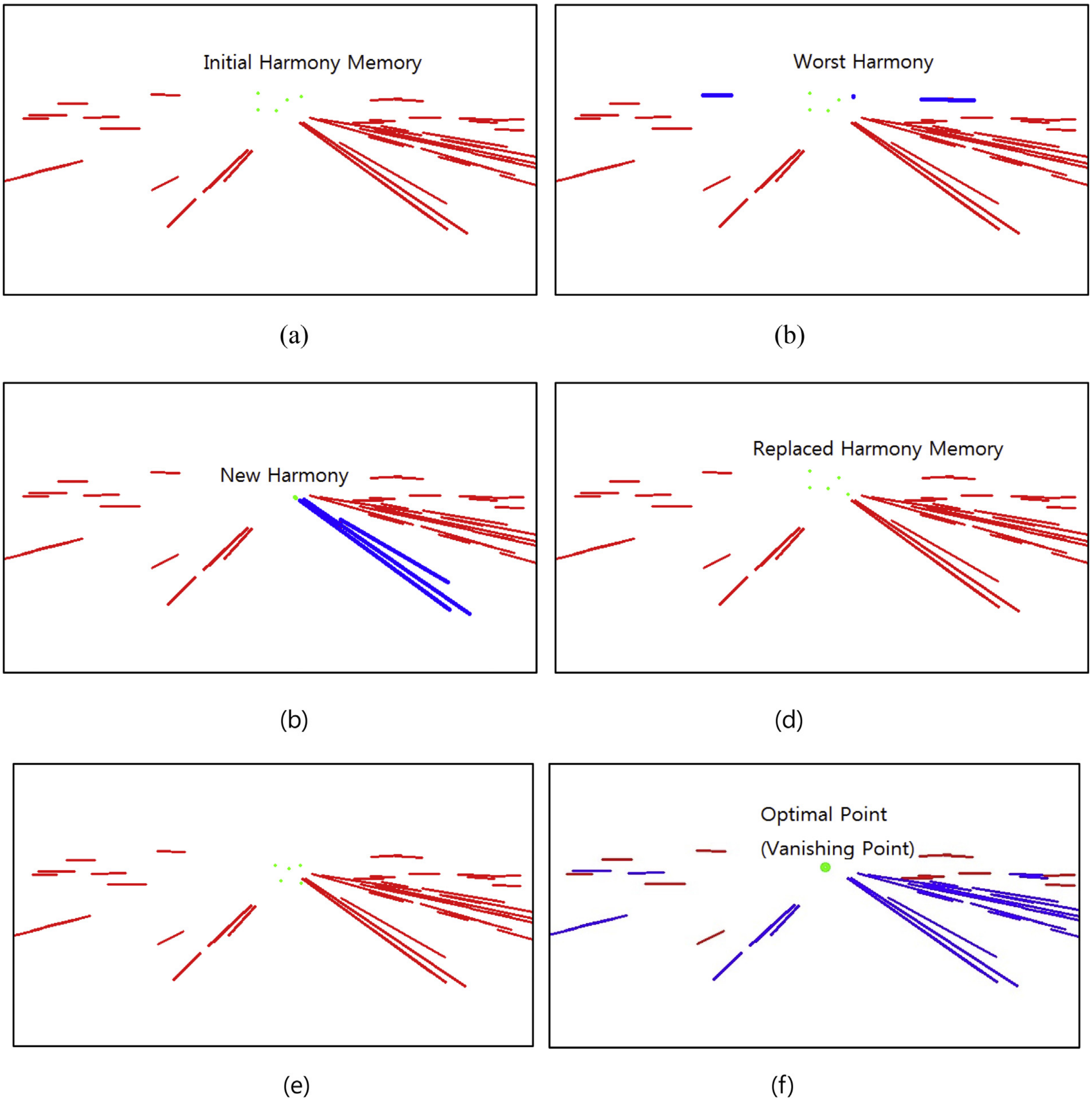
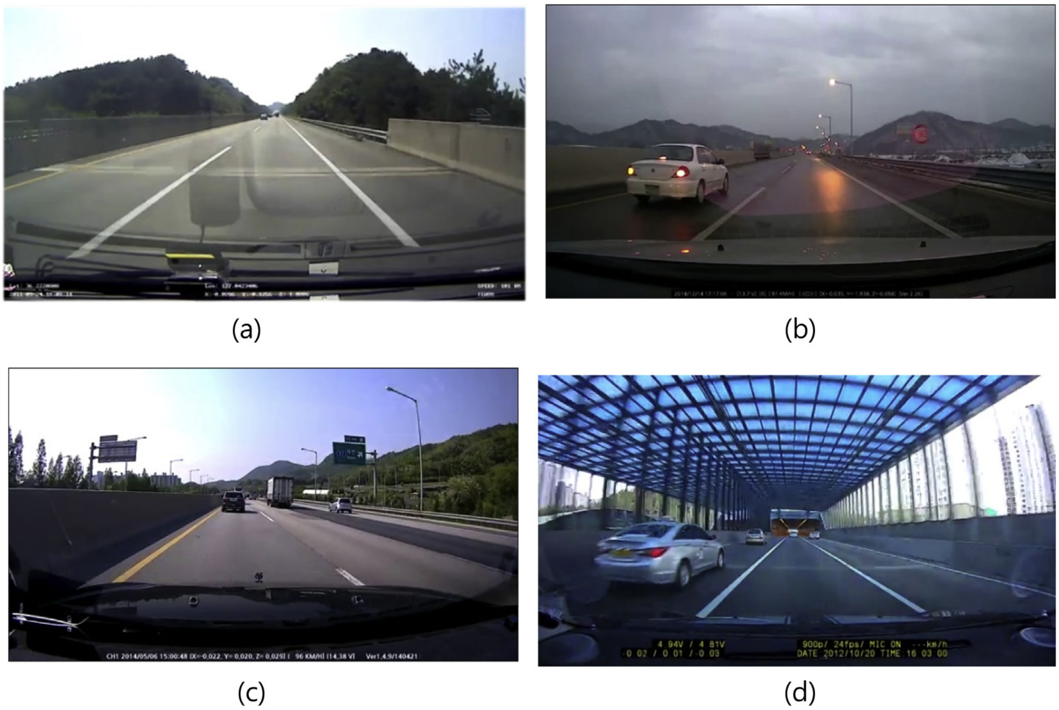


Fig. 7. Process of vanishing point detection by HS algorithm.


(a)

(b)

(c)

(d)

Fig. 8. Experimental video images.

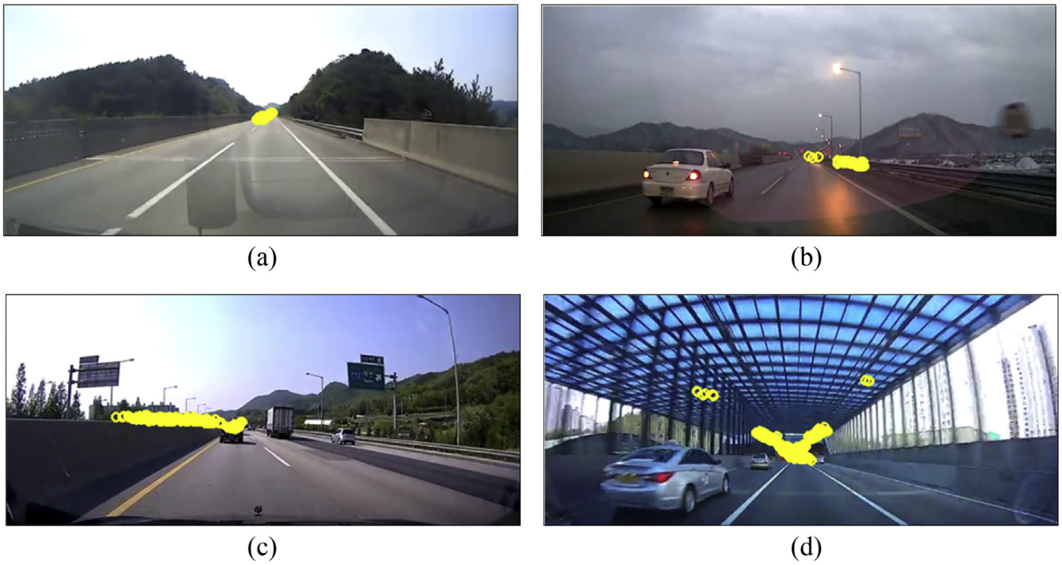
images used for the experiments is 640 × 360. The images were filmed in front of the vehicles, using a dashboard camera.

For this real time computation, we used several video clips, which have 24 frames per second (fps) or 29 fps. Hence, execution time should be less than 1/24 s (= 41.7 ms (ms)) or 1/29 s (= 34.5 ms) for each frame. Actually, both RANSAC-based and HS-based computations took around 31 ms for each frame. Thus, there was no problem to process serial images. For better understanding of smooth image flow, we provide two video clips [35,36].

Fig. 8 shows the images used for experiments under various road conditions: the image in Fig. 8-(a) shows ideal road condition; Fig. 8-(b) shows the condition in the evening; Fig. 8-(c) shows many noises; and Fig. 8-(d) shows the condition in a tunnel.

Fig. 9 shows an example of assumed vanishing points in different locations: Fig. 9-(a) shows the result using RANSAC for the 293rd frame of Fig. 8-(a) image; and similarly, Fig. 9-(b) to 9-(d) are the test results of Fig. 8-(a) to 8-(d) using RANSAC after 10,000 trials.

Fig. 10 shows the results of HS which returns more stable results almost every time under the identical trials (10,000 times). The yellow-colored circle shows the location of detected vanishing point.


(a)

(b)

(c)

(d)

Fig. 9. Detected vanishing points by RANSAC method.

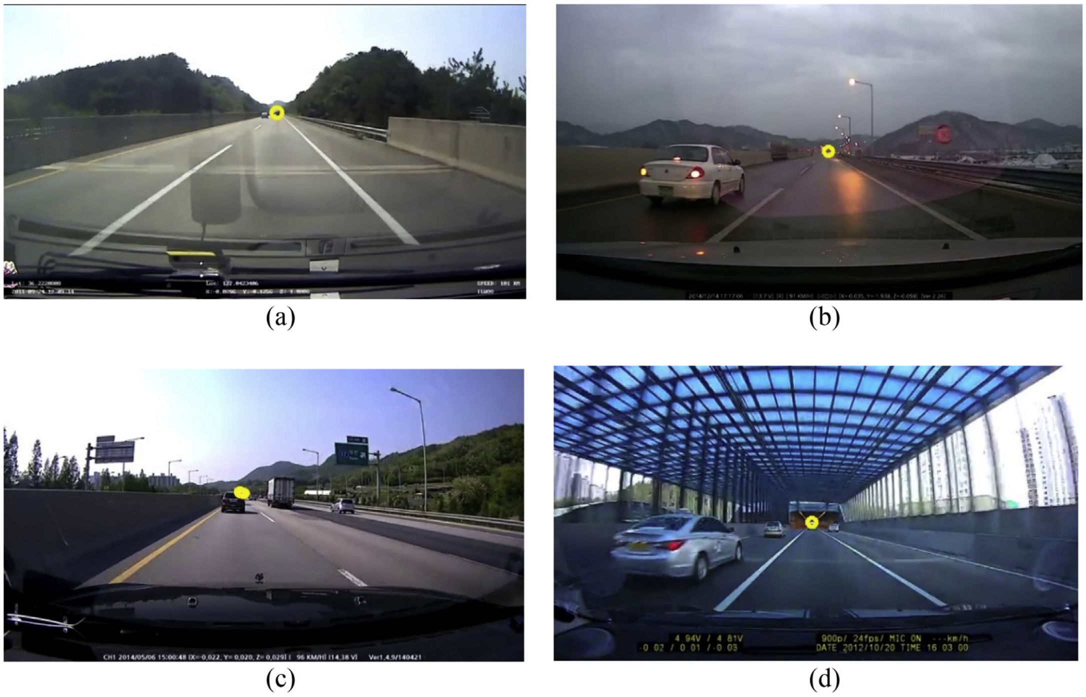


Fig. 10. Detected vanishing points by HS algorithm.

Table 1
Comparison of performance between RANSAC and HS methods.

Method	Try Number	Road (a)	Road (b)	Road (c)	Road (d)
RANSAC	Image	Fig. 9-(a)	Fig. 9-(b)	Fig. 9-(c)	Fig. 9-(d)
	Try 1	0.12	3.04	8.75	15.01
	Try 2	0.15	3.18	8.26	15.22
	Try 3	0.13	3.05	8.44	14.71
	Try 4	0.14	3.11	8.19	14.83
	Try 5	0.17	3.08	8.77	14.98
	Average	**0.14**	**3.09**	**8.78**	**14.95**
HS	Image	Fig. 10-(a)	Fig. 10-(b)	Fig. 10-(c)	Fig. 10-(d)
	Try 1	0.08	0.09	0.09	0.10
	Try 2	0.09	0.07	0.08	0.09
	Try 3	0.07	0.08	0.10	0.09
	Try 4	0.08	0.09	0.09	0.08
	Try 5	0.09	0.09	0.10	0.09
	Average	**0.086**	**0.084**	**0.092**	**0.090**

When the vanishing point is detected by repeatedly testing the same image 10,000 times using RANSAC and HS techniques, the absolute value of covariance computed by the average coordinates of vanishing points is shown in Table 1. This reflects that certain amount of error is included in the resulting vanishing points. Fig. 9-(a) shows that the absolute value of the covariance is 0.14. Because the figure is in ideal condition, the error amount is not so big. Meanwhile, the image in Fig. 9-(d) is a case where a number of errors exist in the detection of the vanishing point, and the absolute value of the covariance is 14.95. However, as shown in Fig. 10, almost all images have far less covariance, which means that the result from HS is fairly stable, showing almost no critical errors.

We think that the instability of the RANSAC algorithm in detecting vanishing points comes from its random nature. Every iteration it generates totally random N hypotheses. Thus, there is a chance of skewed population generation. Meanwhile, the HS algorithm utilizes previously conserved HM as N hypotheses. Thus, there is much lower chance of skewed population generation.

When we considered additional images, we obtained similar results where HS-based approach stably found more accurate vanishing points than RANSAC-based approach, as shown in Fig. 11.

We further tried to compare HS with an improved RANSAC algorithm, named MLESAC (Maximum Likelihood Estimation Sample Consensus) [32], the results of MLESAC (the average absolute value of the covariance for Case (a) image is 0.13; that for Case (b) image is 2.97; that for Case (c) image is 8.36; and that for Case (d) image is 13.13) were slightly improved from those of RANSAC. However, the results of HS are still much better than those of MLESAC.

5. Conclusions

This study proposes a vanishing point detection model using a HS algorithm, which hugely reduced the error found in existing RANSAC algorithm. While the RANSAC algorithm detects much dispersed area as vanishing points, the HS algorithm could pinpoint the area of vanishing points more stably.

Statistical analysis using covariance between each point and averaged point also supported this fact. While the RANSAC method has the covariance in the range 0.14–14.95, the HS algorithm has 0.084 to 0.092. Thus, it proves the fact that the proposed HS method, when concerning vanishing point detection, produces more accurate detection results with fewer errors than the existing RANSAC method. Because the proposed method is able to stably detect the vanishing point, we hope this model to be used for real-world self-driving car to detect road lanes. Also, we want to test this model in harsher environment of bad weather, circular path, or non-highway in the future.

Recently, machine learning techniques like convolution neural network (CNN) have been applied to vanishing point detection [31]. When compared with HS-based approach, CNN-based approach requires extensive pre-process of 63,196 image training and heavier system specification (Quad Core CPU, 16 GB RAM, additional GPU). Moreover, CNN-based approach just found approximate location of vanishing point out of 100–900 candidate locations while HS-based approach could pinpoint the exact location out of 230,400 pixels. It appears that HS-based approach currently provides more accurate and faster detection for real-time road images. Nonetheless, we also would like to consider

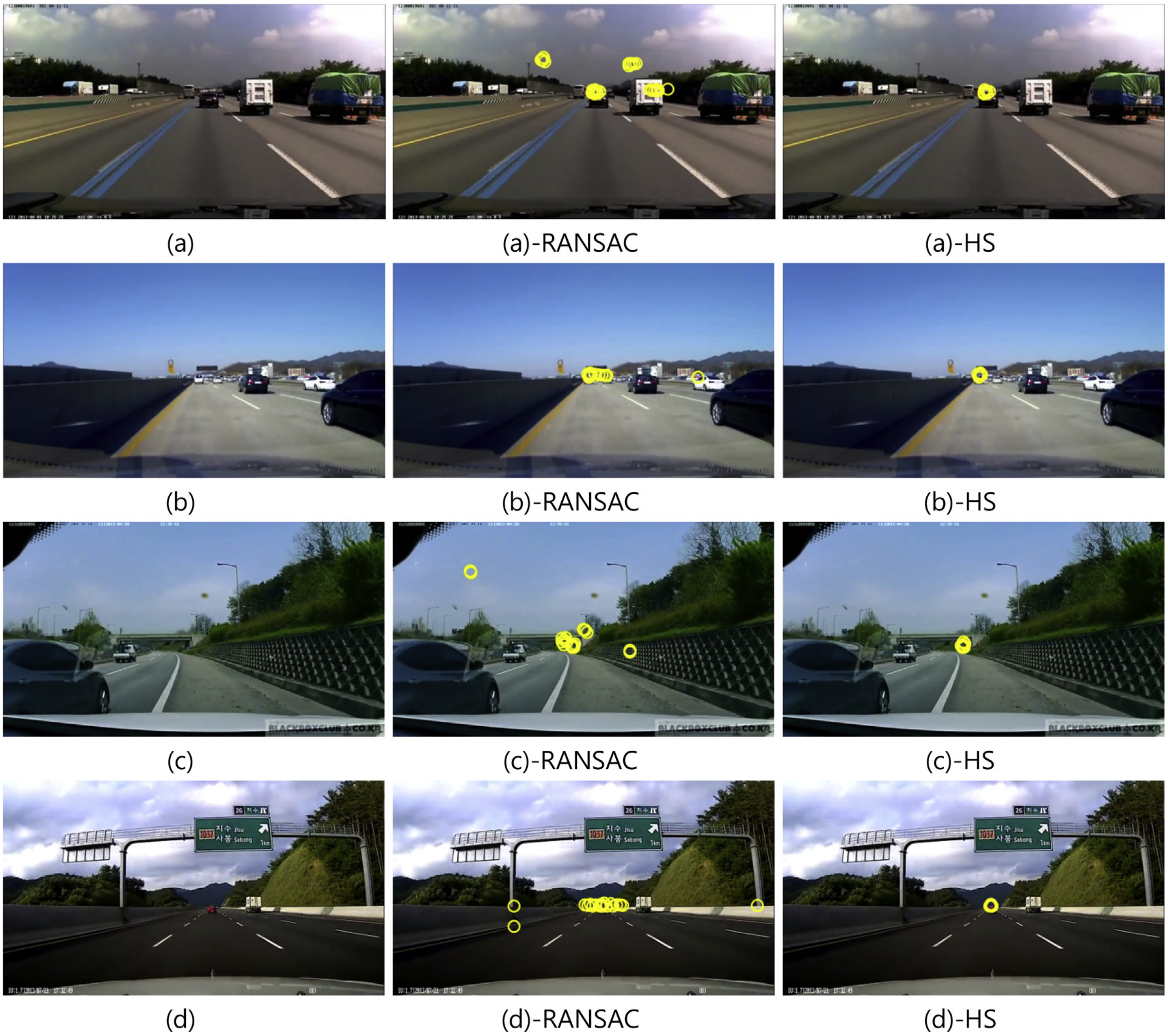


Fig. 11. Additional examples of vanishing point detection.

the CNN technique if it could enhance detection quality in the future.

Conflicts of interest

The authors declare that there is no conflict of interests regarding the publication of this article.

Acknowledgements

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC support program(IITP-2018-2017-0-01630) supervised by the IITP(Institute for Information & communications Technology Promotion).

References

[1] H.T. Kim, A vanishing point detection method based on the empirical weighting of the lines of artificial structures, J. Korean Inst. Inf. Sci. Eng. 42 (no. 5) (2015) 642–651.
[2] L. Zhang, R. Koch, Vanishing points estimation and line classification in a manhattan world, in: Proceedings of the Asian Conference on Computer Vision, 2012, pp. 38–51.
[3] C. Rasmussen, Grouping dominant orientations for Ill-structured road following, in: Proceeding of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, 2004 pp.I-470-I-477.
[4] S.T. Banard, Interpreting perspective images, Artif. Intell. 21 (no. 4) (1983) 435–462.
[5] S. Lim, D. Lee, Y. Park, Lane detection and tracking using classification in image sequences, Trans. Internet Inf. Syst. 8 (no. 12) (2014) 4489–4501.
[6] A.B. Hillel, R. Lerner, D. Levi, G. Raz, Recent progress in road and lane detection: a survey, Mach. Vis. Appl. 25 (no. 3) (2014) 727–745.
[7] Y. Wang, E.K. Teoh, D. Shen, Lane detection and tracking using B-snake, Image Vis Comput. 22 (4) (2004) 269–280.
[8] C. Rother, A new approach to vanishing point detection in architectural environments, Image Vis Comput. 20 (9) (2002) 647–655.
[9] Y. Xu, S. Oh, A. Hoogs, A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1376–1383.
[10] H.J. Liu, A fast method for vanishing point estimation and tracking and its application in road images, in: Proceedings of the 6th International Conference on its Telecommunications, 2006, pp. 106–109.
[11] T. Suttorp, T. Bucher, Robust vanishing point estimation for driver assistance, in: Proceedings of the IEEE ITSC 2006, 2006, pp. 1550–1555.
[12] P. Moghadam, J.A. Starzyk, W.S. Wijesoma, Fast vanishing point detection in unstructured environments, IEEE Trans. Image Process. 21 (no. 1) (2012) 425–430.
[13] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2003.
[14] V. Cantoni, L. Lombardi, M. Porta, N. Sicard, Vanishing point detection: representation analysis and new approaches, in: Proceedings of 11th International Conference on Image Analysis and Processing, 2001, pp. 90–94.
[15] F. Stentiford, Attension-based vanishing point detection, in: Proceedings of the 2006 IEEE International Conference on Image Processing(ICIP), 2006, pp. 417–420.
[16] H. Heo, G.T. Han, A robust real-time lane detection for road with slope, KIPS Trans. Software Data Eng. 2 (6) (2013).

[17] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, Cengage Learning, 2014.

[18] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (no. 6) (1981) 381–395.

[19] Z.W. Geem, J.H. Kim, G.V. Loganathan, A New heuristic optimization algorithm: harmony search, Simulation 76 (2) (2001) 60–68.

[20] K.S. Lee, Z.W. Geem, S.H. Lee, K.W. Bae, The harmony search heuristic algorithm for discrete structural optimization, Eng. Optim. 37 (no. 7) (2005) 663–684.

[21] O. Moh'd Alia, R. Mandava, M.E. Aziz, A hybrid harmony search algorithm for MRI brain segmentation, Evol. Intell. 4 (1) (2011) 31–49.

[22] J. Fourie, S. Mills, R. Green, Harmony filter: a robust visual tracking system using the improved harmony search algorithm, Image Vis Comput. 28 (no. 12) (2010) 1702–1716.

[23] D. Manjarres, I. Landa-Torres, S. Gil-Lopez, J. Del Ser, M.N. Bilbao, S. Salcedo-Sanz, Z.W. Geem, A survey on applications of the harmony search algorithm, Eng. Appl. Artif. Intell. 26 (8) (2013) 1818–1831.

[24] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, Comput. Meth. Appl. Mech. Eng. 194 (36) (2005) 3902–3933.

[25] J.H. Kim, Z.W. Geem, E.S. Kim, Parameter estimation of the nonlinear Muskingum model using harmony search, J. Am. Water Resour. Assoc. 37 (5) (2001) 1131–1138.

[26] E. Imre, A. Hilton, Order statistics of RANSAC and their practical application, Int. J. Comput. Vis. 111 (3) (2014) 276–297.

[27] X. Wu, Q. Zhao, W. Bu, A SIFT-based contactless palm print verification approach using iterative RANSAC and local palm print descriptors, Pattern Recogn. 47 (no. 10) (2014) 3314–3326.

[28] F. Zhou, Y. Cui, Y. Wang, L. Liu, H. Gao, Accurate and robust estimation of camera parameters using RANSAC, Optic Laser. Eng. 51 (3) (2013) 197–212.

[29] J. Matas, O. Chum, Randomized RANSAC with Td,d test, Image Vis Comput. 22 (10) (2004) 837–842.

[30] C.-M. Cheng, S.-H. Lai, A consensus sampling technique for fast and robust model fitting, Pattern Recogn. 42 (7) (2009) 1318–1329.

[31] A. Borji, Vanishing point detection with convolution neural networks, in: Proceedings of Scene Understanding Workshop (SUNw 2016), 2016.

[32] P.H. Torr, A. Zisserman, MLESAC: a new robust estimator with application to estimating image geometry, Comput. Vis. Image Understand. 78 (1) (2000) 138–156.

[33] K. Pearson, On lines and planes of closest fit to systems of points in space, Phil. Mag. Series 6 (11) (1901) 559–572, 2.

[34] R. Collins, R. Weiss, Vanishing point calculation as a statistical inference on the unit sphere, in: Proceedings of International Conference on Computer Vision, 1990.

[35] https://sites.google.com/a/hydroteq.com/www/VP_RANSAC.wmv?attredirects=0&d=1.

[36] https://sites.google.com/a/hydroteq.com/www/VP_HS.wmv?attredirects=0&d=1.