

Traffic-sign Recognition for Visually Impaired Pedestrians in Kyrgyzstan: Two-keypoint SIFT/BRISK Descriptor with CameraX

Ayman Aljarbouh¹, Dmytro Zubov^{1,*}, Andrey Kupin² and Nurlan Shaidullaev¹

¹ University of Central Asia, 125/1 Toktogul Street, Bishkek, 720001, Kyrgyzstan

² Kryvyi Rih National University, 11 Vitaly Matusevich, Kryvyi Rih, 50027, Ukraine

Abstract

The traffic-sign recognition system developed in this study aims to assist the spatial cognition and mobility of visually impaired pedestrians in Kyrgyzstan. The system employs a two-keypoint binary descriptor that implements the BRISK algorithm to find sampling patterns on the image. Pairs of keypoints are localized using the SIFT method. The developed Java Android mobile application implements the SIFT and BRISK approaches in real-time on Android CameraX using AdaBoost classifiers and multithreading. With a knowledge base of 86 sampling patterns, the execution time is 0.1 s for an example with the traffic sign "Crosswalk left". In experiments conducted at distances 1.5 m to 3.5 m in the city of Naryn, Kyrgyzstan, the presented SIFT/BRISK detector demonstrated a true negative of 100 % and a true positive close to 100 % (Blackview BV6600 Pro and Doogee S96 Pro smartphones achieved 100 % and 75 %, respectively) rates at 3.5 m. This pilot project is expected to continue with more precise image descriptors for longer distances.

Keywords

Two-keypoint descriptor, visually impaired, SIFT, BRISK, Android CameraX

1. Introduction

The visually impaired and blinds (VIBs) have made significant progress in social integration over the last five decades. This achievement is mostly based on inclusive smart technologies that create synergies between the community and VIBs [1]. Despite numerous assistive mobile applications (e.g., BDS (BeiDou Navigation Satellite System) WeChat and Google Maps) for spatial cognition, VIB navigation [2] remains problematic for the last mile, such as finding the entrance and identifying traffic signs [3-11].

In this study, a Java Android mobile application was developed to detect and recognize Kyrgyz traffic signs using the SIFT and BRISK (Scale-Invariant Feature Transform and Binary Robust Invariant Scalable Keypoints) methods [12, 13] to localize keypoints and find sampling patterns with a two-keypoint binary descriptor, respectively. The true positive (i.e., recognition accuracy) and true negative (i.e., crucial mistakes) rates [14] are expected to be near 100 % at distances from 1.5 m to 3.5 m from the traffic sign.

To support VIBs, a new mobile application was developed to recognize traffic signs in Kyrgyzstan. Two key problems were solved in this study:

1. A novel technique has been applied for image processing. In the preprocessing step, the method `Bitmap.createScaledBitmap` creates a new bitmap scaled to a maximum resolution of 500 pixels with bilinear filtering. From up to four hundred keypoints detected by the SIFT method, two keypoints are selected. Then, the BRISK binary two-keypoint descriptor is

COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine

* Corresponding author

✉ ayman.aljarbouh@ucentralasia.org (A. Aljarbouh); dzubov@ieeee.org (D. Zubov); kupin@knu.edu.ua (A. Kupin); nurlan.shaidullaev@ucentralasia.org (N. Shaidullaev)

🆔 0000-0002-3909-2227 (A. Aljarbouh); 0000-0002-5601-7827 (D. Zubov); 0000-0001-7569-1721 (A. Kupin); 0009-0003-5165-897X (N. Shaidullaev)



© 2024 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

designed employing a 291-point shape. The Hamming distance threshold is 19600 after five AdaBoost weak classifiers, which shows a true positive rate close to 100 % (smartphone Blackview BV6600 Pro – 100 %, Doogee S96 Pro – 75 % at 3.5 m) and a true negative rate of 100 % at distances from 1.5 m to 3.5 m.

2. A multithreaded Java Android application was developed using the CameraX library [15]. The image, smartphone soft-/hardware, and number of keypoints have an effect on the execution time. In the experiment with the traffic sign “Crosswalk left”, the smartphone Doogee S96 Pro takes 0.1 s to find the sampling pattern using the knowledge base with 86 elements.

2. Related Works

The World Health Organization showed that over 2.2 billion people experience vision impairment worldwide in 2021 [10], and hence assistive tools are in continuous demand. Traffic-sign recognition systems in cars [3] have been widely proposed for the market. The leading approach is based on convolutional neural networks and specific datasets, e.g., Tunisian traffic signs [4]. The percentage of wrongly recognized signs can reach 25 % [3], which is unacceptable for VIBs. Analysis of existing commercial products for VIBs, such as those referenced in [5-11], shows that they do not support the recognition of traffic signs related to pedestrians. Hence, the development of a mobile application that includes this functionality is a crucial task that should be undertaken to support the VIBs navigation near roads. Moreover, the usage of existing technologies is reasonable since it speeds up the development process, as was done in this study. The distance between the VIB and the traffic sign is assumed to be up to 4 m, which is the estimated width of the pedestrian path. The analyzed traffic signs are supposed to be of good quality and produced according to state standards.

Google’s Android platform has been taking over 70 % of the market share last five years. The CameraX Android API (application programming interface) is a Google Android native approach to work with different cameras on Android smartphones. CameraX is a Jetpack support library, which is considered the easiest way to make the Android camera application.

3. Methods

3.1. Architecture of two-keypoint SIFT detector and BRISK descriptor with CameraX Android API

Two-keypoint SIFT detector and BRISK descriptor with CameraX Android API employ the method presented in [16]. It and consists of three steps (see Figure 1):

1. Keypoints localization: The SIFT method is used to localize keypoints on the template image.
2. Image descriptor design: The BRISK method is employed to design image descriptors using pairs of keypoints that are empirically determined by an expert. This procedure will be automated in the future.
3. Image matching: Image capturing is performed using an Android CameraX library.

The image is then downsampled with bilinear filtering in the method *Bitmap.createScaledBitmap* and converted to grayscale from RGB (Red-Green-Blue) format using the luminosity function [17]. The best pairs of keypoints calculated by the SIFT algorithm are selected for the design of the BRISK binary descriptor. Image matching uses AdaBoost classifiers and Hamming distance [18] (see Figure 2). Figure 3 presents two flowcharts: one for keypoints localization with the SIFT method and image descriptor design with the BRISK method (on the left), and another for the image matching algorithm (on the right).

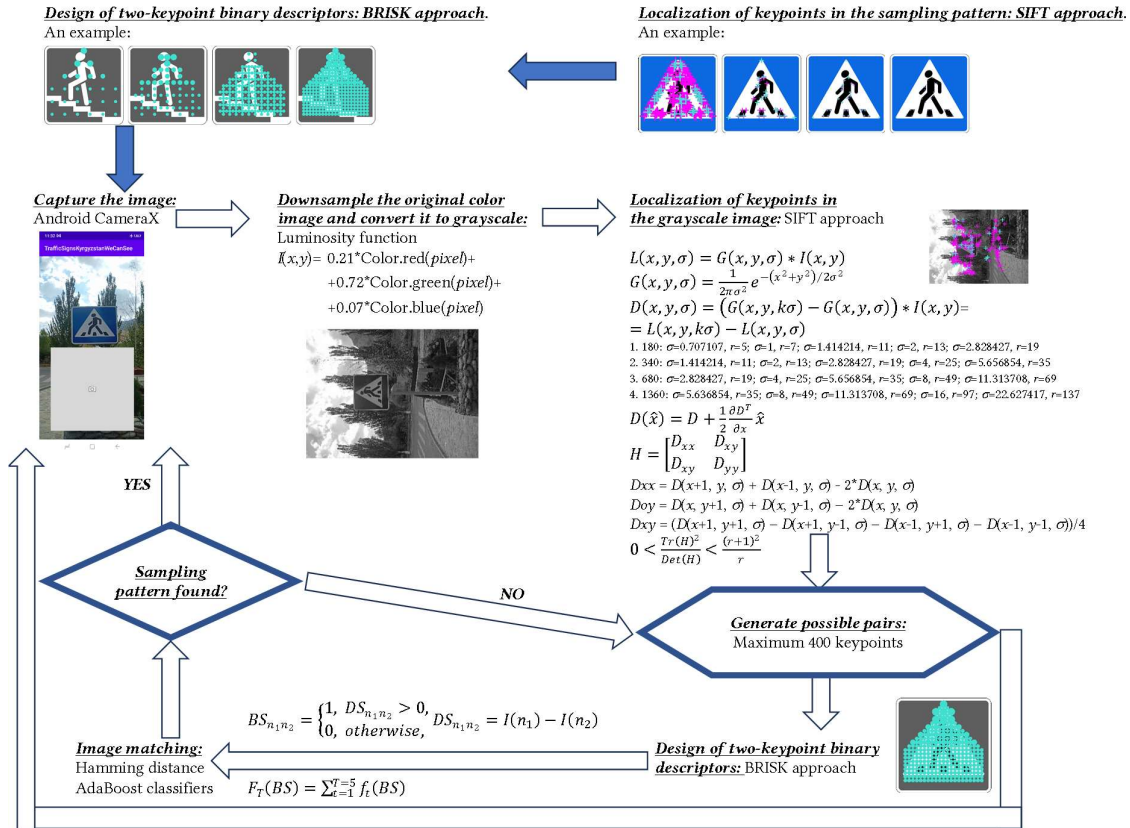


Figure 1: Two-keypoint SIFT detector and BRISK descriptor with CameraX Android API: A diagram

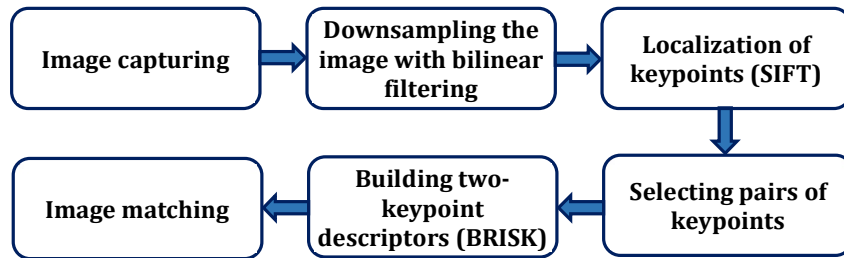


Figure 2: The architecture of the proposed image matching

3.2. Classification of Kyrgyz traffic signs for pedestrians

As of August 2023, Kyrgyzstan had over 200 road signs [19], including 13 related to pedestrians (see Table 1).

The crosswalk signs “Crosswalk left”, “Crosswalk right”, and “Zebra crossing” are combined into a group “Crosswalk”, as well as the signs “Emergency exit left/right” into “Emergency exit”.

3.3. SIFT keypoint localization

Regarding the SIFT keypoint localization, the AdaBoost cascade classifier is employed in this study. In this approach, scale-invariant locations of keypoints are searched in different scales. The convolution of two-dimensional Gaussian function $G(x,y,\sigma)$ and input grayscale image $I(x,y)$ gives a filtered image:

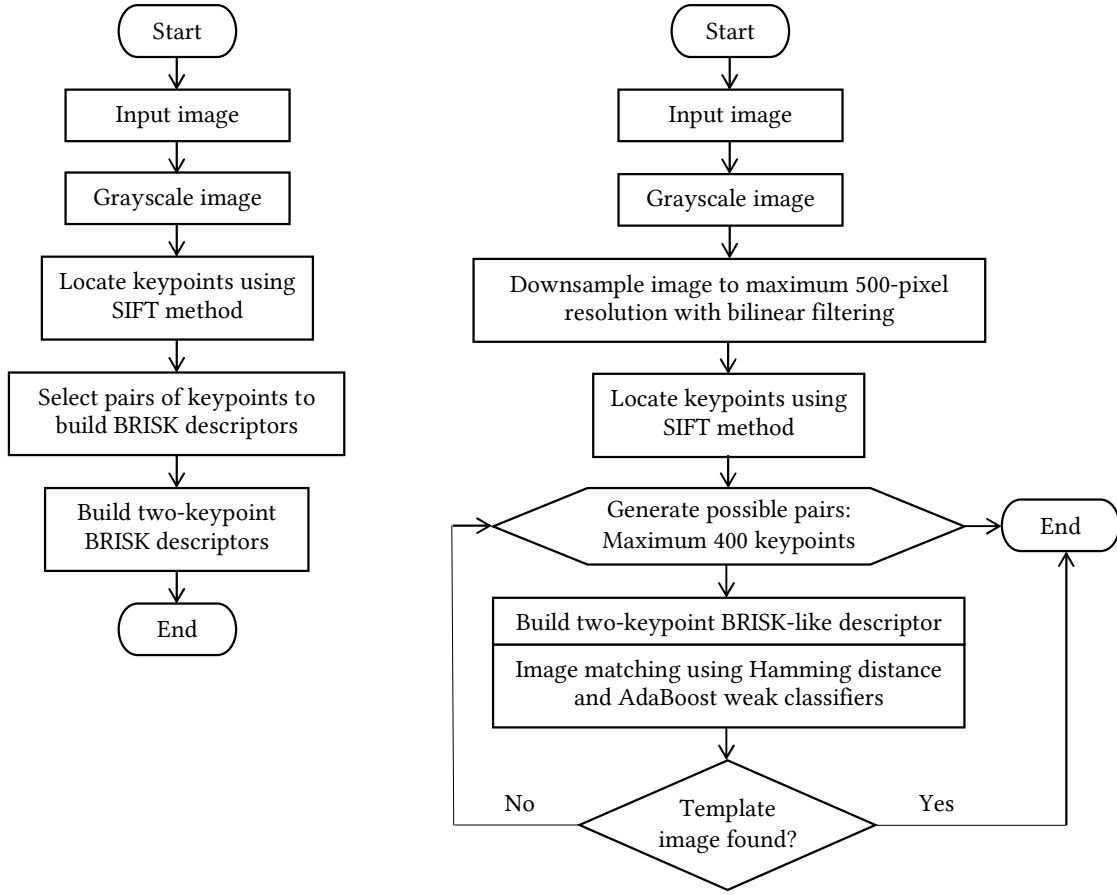


Figure 3: Flowcharts of the keypoints localization with the SIFT method and the image descriptor design with the BRISK method (left flowchart) and the image matching algorithm (right flowchart)

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \quad (1)$$

where '*' is the convolution operation, σ is the population standard deviation, x and y are the pixel coordinates, and a Gaussian function:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \quad (2)$$

In this study, the luminosity function [17] converts the image from RGB to grayscale $I(x, y)$ using Android class *Color*:

$$I(x, y) = Color.red(pixel) * 0.21 + Color.green(pixel) * 0.72 + Color.blue(pixel) * 0.007, \quad (3)$$

where *Color.red*, *Color.green*, and *Color.blue* are Java methods, *pixel* is the smallest element that can be addressed in a raster RGB image.

The DoG (Difference of Gaussians) function $D(x, y, \sigma)$ is employed to find keypoints that are stable across different scales. DoG is the result of subtracting two neighbour scales that are smoothed by Gaussian filters with a different weight k :














$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma). \quad (4)$$

The DoG function is the approximation of the scale-normalized Laplacian of Gaussian $\sigma^2 \nabla^2 G$. Building DoG $D(x, y, \sigma)$ follows the method proposed in [16]. To generate five SIFT scales, four images (maximum dimension sizes are 180, 340, 680, and 1360 pixels, i.e., four SIFT octaves) are smoothed five times in the Gaussian blur operator with five square matrix orders r and population standard deviations σ :

1. 180: $r=5, \sigma=0.707107$; $r=7, \sigma=1$; $r=11, \sigma=1.414214$; $r=13, \sigma=2$; $r=19, \sigma=2.828427$.

2. 340: $r=11, \sigma=1.414214$; $r=13, \sigma=2$; $r=19, \sigma=2.828427$; $r=25, \sigma=4$; $r=35, \sigma=5.656854$.
3. 680: $r=19, \sigma=2.828427$; $r=25, \sigma=4$; $r=35, \sigma=5.656854$; $r=49, \sigma=8$; $r=69, \sigma=11.313708$.
4. 1360: $r=35, \sigma=5.636854$; $r=49, \sigma=8$; $r=69, \sigma=11.313708$; $r=97, \sigma=16$; $r=137, \sigma=22.627417$.

Table 1
Kyrgyz traffic signs related to pedestrians

| No. | Designation | Traffic sign icon | No. of sampling patterns |
|-----|----------------------------------|---|--------------------------|
| 1 | Above ground pedestrian crossing |  | 6 |
| 2 | Bike crossing |  | 7 |
| 3 | Bike path |  | 13 |
| 4 | Bus stop |  | 1 |
| 5 | Crosswalk left |  | 16 |
| 6 | Crosswalk right |  | 1 |
| 7 | Emergency exit left |  | 7 |
| 8 | Emergency exit right |  | 8 |
| 9 | No entry for pedestrians |  | 10 |
| 10 | Pedestrian path |  | 10 |
| 11 | Tram stop |  | 5 |
| 12 | Underground pedestrian crossing |  | 1 |
| 13 | Zebra crossing |  | 1 |

Detection of keypoint candidates (i.e., local maxima and minima of DoG function) is similar to the presented in [16, 20] methodology. Keypoints are rejected using the Taylor expansion of $D(x, y, \sigma)$:

$$D(x) = D + \frac{\partial D^T}{\partial x} + \frac{1}{2} x^T \frac{\partial^2 D^T}{\partial x^2} x, \quad (5)$$

where $\mathbf{x}=(x, y, \sigma)^T$ is the offset from and D and its derivatives are computed at a particular point. To find the extremum \hat{x} , the equation $D'(\mathbf{x}) = 0$ should be solved ($D'(\mathbf{x})$ is the derivative of D with respect to \mathbf{x}):

$$\hat{x} = \frac{\partial^2 D^{-1} \partial D}{\partial x^2 \partial x}. \quad (6)$$

Unstable extrema are rejected if $|D(\hat{x})| < 0.03$. The extremum $D(\hat{x})$ can be found by combining Eq. (6) and Eq. (5):

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x}. \quad (7)$$

In this study, edges are detected employing a 2×2 Hessian matrix [12, 20]:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}, \quad (8)$$

where derivatives D_{xx} , D_{xy} , and D_{yy} are as follows:

$$D_{xx} = D(x+1, y, \sigma) + D(x-1, y, \sigma) - 2 * D(x, y, \sigma), \quad (9)$$

$$D_{yy} = D(x, y+1, \sigma) + D(x, y-1, \sigma) - 2 * D(x, y, \sigma), \quad (10)$$

$$D_{xy} = (D(x+1, y+1, \sigma) - D(x+1, y-1, \sigma) - D(x-1, y+1, \sigma) - D(x-1, y-1, \sigma)) / 4. \quad (11)$$

To eliminate the number of keypoints, the following inequality should be satisfied [12, 16, 20]:

$$0 < \frac{Tr(H)^2}{Det(H)} < 12.1. \quad (12)$$

3.4. Two-keypoint BRISK descriptor design

In this study, the BRISK algorithm employs a binary descriptor with 291 points to depict the template image. The orientation and scale of the sample pattern are calculated using the positions of two keypoints. In the present software version, a human expert chooses two keypoints by examining keypoints with a consistent location at various octaves.

In the binary descriptor, 291 points are split as follows: 0-24 (1st group), 25-82 (2nd group), and 83-290 (3rd group). Figure 4 shows an example of the descriptor for the traffic sign “Above ground pedestrian crossing” (greyscale representation): (A) – points 0-24, (B) – 25-82, (C) – 83-290, (D) – 0-290. The distance between any two points is computed via the Euclidean distance between two keypoints: for the 1st group point on the line connecting two keypoints, the distance to any nearest point is one-third of the distance between two keypoints. Each point n is associated with the mean pixel intensity $I(n)$ in a circle of a radius $1/24$, $1/16$, or $1/12$ of the Euclidean distance $E_{n_1 n_2}$ between two keypoints n_1 and n_2 [16].

The Hamming distance is calculated via the binary string $BS_{n_1 n_2}$, which is based on the comparison of average pixel intensities $I(n_1)$ and $I(n_2)$ at points n_1 and n_2 , respectively:

$$BS_{n_1 n_2} = \begin{cases} 1, & DS_{n_1 n_2} > 0, \\ 0, & otherwise, \end{cases} \quad (13)$$

where the string $DS_{n_1 n_2}$ is as follows:

$$DS_{n_1 n_2} = I(n_1) - I(n_2). \quad (14)$$

For a specific point n_1 , absolute differences $abs(DS_{n_1 n_2})$ are descending sorted within the appropriate group:

1. Points 0-24: the image binary descriptor considers the first 12 absolute differences for each point.
2. Points 25-82: the image binary descriptor considers the first 32 absolute differences for each point.
3. Points 83-290: the image binary descriptor considers the first 100 absolute differences for each point.

Therefore, a total of 22956 absolute differences $abs(DS_{n_1 n_2})$ are considered in the image binary descriptor, which can be calculated as follows:

$$25 * 12 + 58 * 32 + 208 * 100 = 300 + 1856 + 20800.$$

4. Experiment

4.1. Knowledge base design

The knowledge base is stored on the smartphone internal storage and includes the following files [16]: ‘root.txt’; N descriptor files “dN.txt”; “Description.txt”; N audio files “NameN.mp3”.

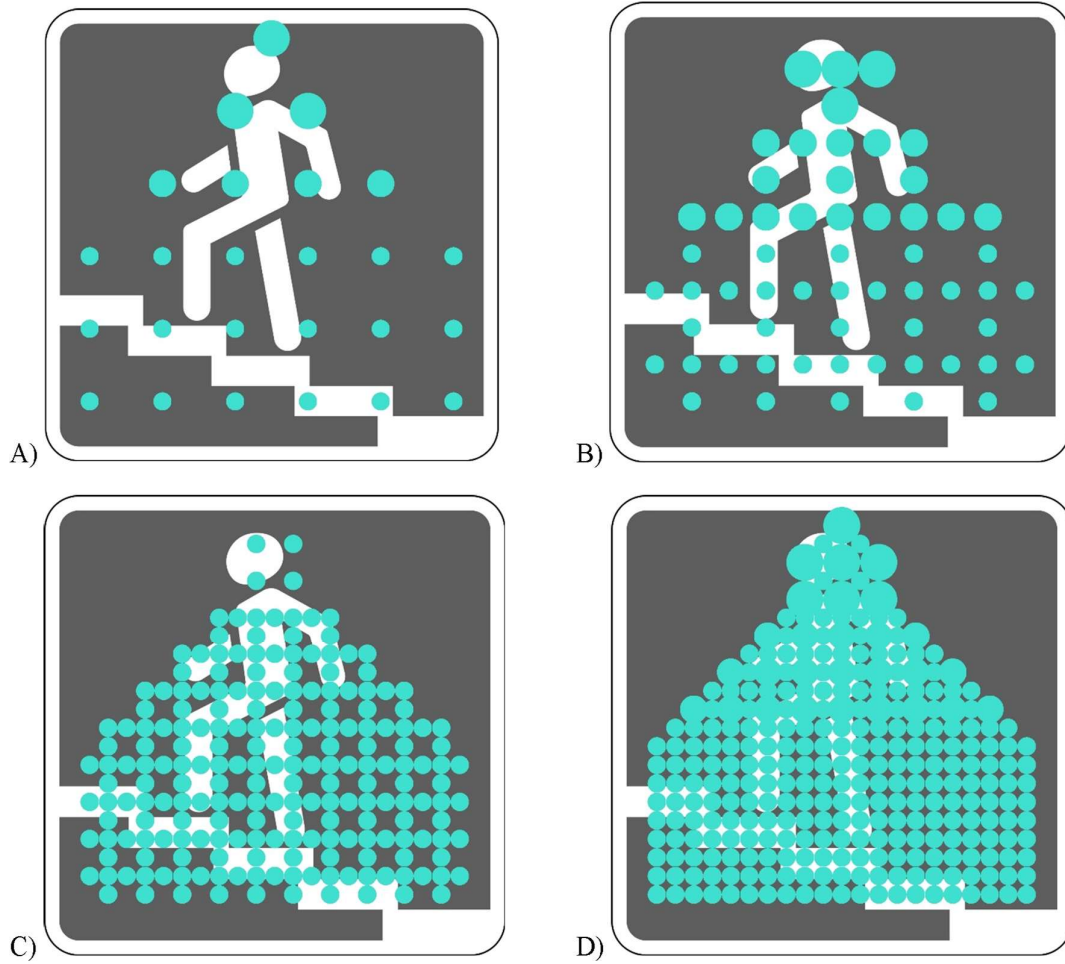


Figure 4: The binary descriptor for the traffic sign “Above ground pedestrian crossing” (greyscale representation): 1st (A), 2nd (B), and 3rd (C) groups, all points (D)

The knowledge base includes 86 sampling patterns ($N=86$; see Table 1) and has a size of 44.1 MB on the internal storage (106 MB in Random Access Memory (RAM) along with other data and code of the Android application), which is available on any up-to-date Android smartphone with operating system (OS) version 10 (10th and 11th are discussed in this study) or newer. The execution time for processing a test image (taken at sunny weather on a campus in Naryn, University of Central Asia, Kyrgyzstan; see Figure 5) on a Doojee S96 Pro smartphone is approximately 0.1 s.

In this study, the Hamming distance measures the similarity between two binary strings $BS_{n_1n_2}$. The image-matching process employs five AdaBoost weak classifiers [21] that use binary decision trees.

4.2. Experiment description

To minimize the execution time of Java Android application, the maximum number of keypoints and the image dimension size are 400 and 500, respectively, which is compatible with any modern camera smartphone since a 2-megapixel sensor captures images of 1600×1200 pixels. To avoid optical distortion effects [22] and locate the binary descriptor within the borders, a new image is created by adding 50-pixel margins to the original image. The method *Bitmap.createScaledBitmap* is used to downsample the original image with bilinear filtering. Then, a greyscale representation is calculated in eight parallel computational threads.



Figure 5: An example of the traffic sign “Crosswalk left” successfully identified during the experiment (photo was taken by smartphone Doozee S96 Pro at sunny weather; author – Dmytro Zubov)

To find keypoints, the image with a maximum dimension size of 500 pixels is smoothed five times. This process generates five scales using three groups (groups are applied sequentially until the target object is detected or not found) of the square matrix orders and population standard deviations in the Gaussian blur operator:

1. $r=7, \sigma=1; r=11, \sigma=1.414214; r=13, \sigma=2; r=19, \sigma=2.828427; r=25, \sigma=4.$
2. $r=11, \sigma=1.414214; r=13, \sigma=2; r=19, \sigma=2.828427; r=25, \sigma=4; r=35, \sigma=5.656854.$
3. $r=5, \sigma=0.5; r=7, \sigma=0.70711; r=9, \sigma=1; r=11, \sigma=1.414214; r=13, \sigma=2.$

To reduce the execution time, four DoG images are calculated employing Eq. (4) in four parallel computational threads.

Keypoints are identified in two parallel computational threads for the third/second/first and fourth/third/second scales. Only 400 keypoints, which are closest to the center of the image according to the Euclidean distance, are considered. Figure 6 presents the scheme of how points are analyzed – the number of side points at the current level is two pixels larger than at the previous one: 1, 3, 5, 7, 9, etc.

The AdaBoost classifier is applied after discarding keypoint pairs whose descriptor points fall beyond the image boundaries:

$$F_T(BS) = \sum_{t=1}^{T=5} f_t(BS), \quad (15)$$

where $f_t(BS)$ is AdaBoost weak classifier and $T=5$ [16]. If the Hamming distance surpasses the threshold 19600, the descriptor is of the template class. The threshold value was selected empirically based on the sum of the outcomes of the five weak classifiers mentioned above. Table 2 summarizes the speedup techniques used in this study.

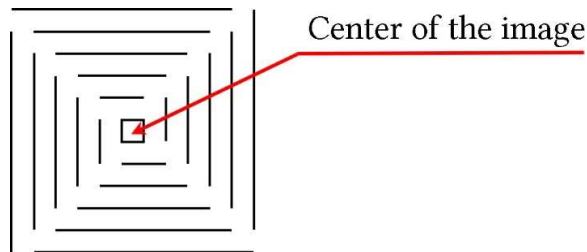


Figure 6: Scheme of the points analysis on the image

Table 2
Speedup techniques in Java Android application

| Operation | Speedup technique |
|--|--------------------------------------|
| Greyscale representation of the color image | Eight parallel computational threads |
| Calculation of DoG images | Four parallel computational threads |
| Localization of keypoints in third/second/first and fourth/third/second scales | Two parallel computational threads |
| Image matching | Five AdaBoost classifiers |

5. Results

In this study, a Java Android application “TrafficSignsKyrgyzstanWeCanSee” employs the proposed method to detect Kyrgyz traffic signs, and hence to support the spatial cognition and mobility of VIBs. Figure 7 presents the screenshot, an original image taken by the smartphone Doogee S96 Pro at cloudy weather (location is a campus in Naryn, University of Central Asia, Kyrgyzstan), and a greyscale image with keypoints (fuchsia and turquoise colors are used for third/second/first and fourth/third/second DoG functions, respectively). Two other Java Android 10 applications (approximately 72 % of Android smartphones can run these applications as of August 2023) were designed in Android Studio 4.0:

1. Localization of all keypoints using SIFT method.
2. Identification of keypoints’ pairs and design of the sample pattern via the BRISK descriptor.

Two smartphones, Doogee S96 Pro and Blackview BV6600 Pro, were used in the experiment. The true positive rate was calculated at different distances from 1.5 m to 5 m in three attempts. The results are shown in Figure 8. Figure 5 presents the photo taken during this experiment. The true positive rate was close to the required 100 % from a distance of 1.5 m to 3.5 m for the traffic sign “Crosswalk left”: it was 100 % for the smartphone Blackview BV6600 Pro and 75 % for the smartphone Doogee S96 Pro at a distance of 3.5 m. The presented two-keypoint SIFT detector and BRISK descriptor showed that the true negative rate was 100 %.

6. Discussion

In this study, a two-keypoint SIFT detector and a BRISK descriptor with CameraX Android API compose the approach to support the navigation and mobility of VIB pedestrians in Kyrgyzstan. In this method, keypoints are localized via the SIFT algorithm, and then selected pairs of keypoints are employed to design the sample pattern, i.e., the binary BRISK descriptor. The image matching is based on the Hamming distance and AdaBoost cascade classifier. The real-life experiment showed test results: a true negative rate is 100 % (this is a crucial parameter for VIBs) and a true positive rate is close to 100 %. In general, the traffic-sign recognition system satisfies requirements and hence can be implemented in practice. However, some elements (e.g., square matrix orders and population standard deviations in the Gaussian blur operator) of the presented approach are empirical, and therefore discussable.

7. Conclusions

In this study, a crucial VIB-assistive software, Java Android mobile application, was developed to recognize Kyrgyz traffic signs using two-keypoint SIFT detector, BRISK descriptor, CameraX Android API, and mp3 audio files to support the spatial cognition of VIBs near roads.

With a knowledge base of 86 sampling patterns, the mobile application shows the real-time performance: the execution time is 0.1 s for example with the traffic sign “Crosswalk left” (location is a campus in Naryn, University of Central Asia, Kyrgyzstan). In experiments on the

distance from 1.5 m to 3.5 m, the presented SIFT/BRISK detector with two-keypoint descriptor showed 100 % true negative rate and a true positive rate close to 100 %: smartphone Blackview BV6600 Pro – 100 %, Doogee S96 Pro – 75 % on the distance 3.5 m.

The real-time performance is achieved using five AdaBoost classifiers for image matching and parallel computational threads for the greyscale representation of the color image (eight threads), calculation of DoG images (four threads), and localization of keypoints (two threads).



Figure 7: An example of the screenshot (A), an original image taken by the smartphone Doogee S96 Pro at cloudy weather (B), and a greyscale image with keypoints (C)

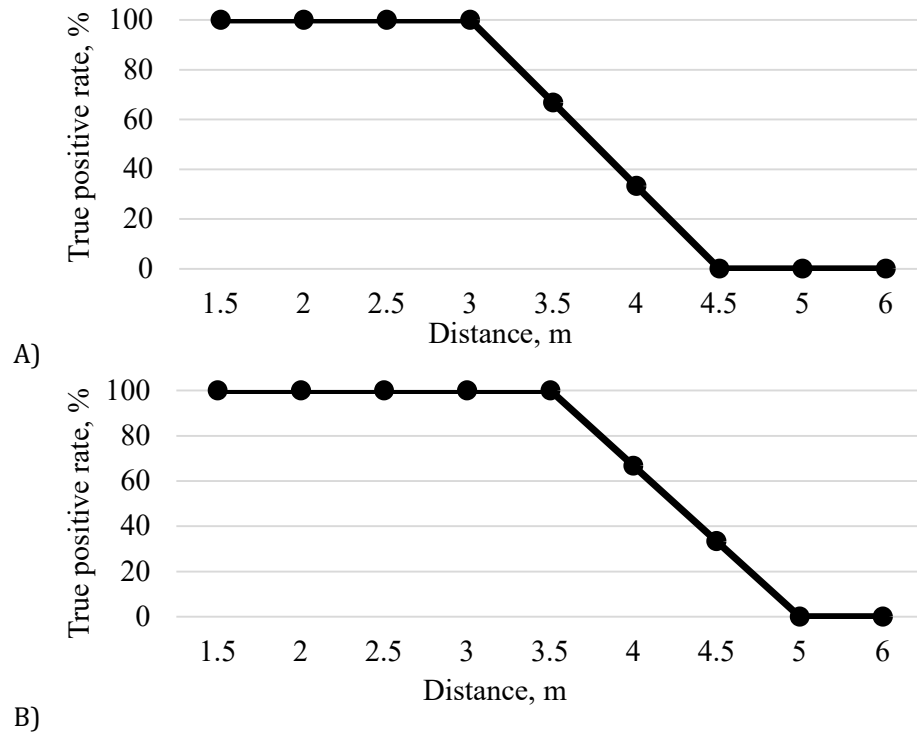


Figure 8: Results of the experiment: true positive rate on different distances for smartphones Dookee S96 Pro (A) and Blackview BV6600 Pro (B)

Analysis of minimum requirements to the hardware shows that the mobile application is runnable on any up-to-date Android camera smartphone because it requires only 44.1 MB on the internal storage (106 MB in RAM along with other data and code) and a 2-megapixel sensor.

The most likely prospect for further development of this study is the design of an image descriptor that is geometrically close to the traffic signs.

8. References

- [1] John Bricout, Paul M. A. Baker, Nathan W. Moon, and Bonita Sharma. "Exploring the Smart Future of Participation: Community, Inclusivity, and People with Disabilities." *International Journal of E-Planning Research* 10.2 (2021): 94-108. doi: 10.4018/IJEPR.20210401.oa8.
- [2] M. Gallay, M. Denis, M. Auvray, Navigation Assistance for Blind Pedestrians: Guidelines for the Design of Devices and Implications for Spatial Cognition, in T. Tenbrink, J. Wiener, C. Claramunt (Eds.), *Representing Space in Cognition: Interrelations of Behaviour, Language, and Formal Models*, Oxford Academic, Oxford, 2013, pp.244-267. doi: 10.1093/acprof:oso/9780199679911.003.0011.
- [3] Darko Babić, Dario Babić, Mario Fiolić, and Željko Šarić. "Analysis of Market-Ready Traffic Sign Recognition Systems in Cars: A Test Field Study." *Energies* 14.12 (2021). doi: 10.3390/en14123697.
- [4] Hana Ben Fredj, Amani Chabbah, Jamel Baili, Hassen Faiedh, and Chokri Souani. "An Efficient Implementation of Traffic Signs Recognition System Using CNN." *Microprocessors and Microsystems* 98 (2023). doi: 10.1016/j.micpro.2023.104791.
- [5] Kanak Manjari, Madhushi Verma, and Gaurav Singal. "A Survey on Assistive Technology for Visually Impaired." *Internet of Things* 11 (2020). doi: 10.1016/j.iot.2020.100188.
- [6] Filippo Amore, Valeria Silvestri, Margherita Guidobaldi, et al. "Efficacy and Patients' Satisfaction with the ORCAM MyEye Device Among Visually Impaired People: A Multicenter Study." *Journal of Medical Systems* 47:11 (2023). doi: 10.1007/s10916-023-01908-5.

- [7] Myneni Madhu Bala, D. N. Vasundhara, Akkineni Haritha, and CH. V. K. N. S. N. Moorthy. "Design, Development and Performance Analysis of Cognitive Assisting Aid with Multi Sensor Fused Navigation for Visually Impaired People." *Journal of Big Data* 10 (2023). doi: 10.1186/s40537-023-00689-5.
- [8] Sonal Mali, Srushti Padade, Swapnali Mote, and Revati Omkar. "An Eye for a Blind: Assistive Technology." *International Research Journal of Engineering and Technology* 3.12 (2016): 532-534.
- [9] Zahra J. Muhsin, Rami Qahwaji, Faruque Ghanchi, and Majid Al-Tae. "Review of Substitutive Assistive Tools and Technologies for People with Visual Impairments: Recent Advancements and Prospects." *Journal on Multimodal User Interfaces* 18 (2023): 135-156. doi: 10.1007/s12193-023-00427-4.
- [10] Adnan Al-Smadi, Talal Al-Qaryouti, Abdurahman Rehan, Homam Assi, and Alhareth Alsharea. A Navigation Tool for Visually Impaired and Blind People, in: *Proceedings of the Eurasia Proceedings of Science, Technology, Engineering & Mathematics*, volume 22 of EPSTEM, ISRES Publishing, Marmaris Turkey, 2023, pp. 119-126. doi: 10.55549/epstem.1338545.
- [11] Matteo Poggi and Stefano Mattocchia. A Wearable Mobility Aid for the Visually Impaired based on Embedded 3D Vision and Deep Learning, in: *Proceedings of the IEEE Symposium on Computers and Communication*, Messina Italy, 2016, IEEE Publishing, pp. 208-213. doi: 10.1109/ISCC.2016.7543741.
- [12] Zetian Tang, Zemin Zhang, Wei Chen, and Wentao Yang. "An SIFT-Based Fast Image Alignment Algorithm for High-Resolution Image." *IEEE Access* 11 (2023): 42012-42041. doi: 10.1109/ACCESS.2023.3270911.
- [13] Guoming Chu, Yan Peng, Xuhong Luo. "ALGD-ORB: An Improved Image Feature Extraction Algorithm with Adaptive Threshold and Local Gray Difference." *PLoS ONE* 18.10 (2023). doi: 10.1371/journal.pone.0293111.
- [14] Alaa Tharwat. "Classification Assessment Methods." *Applied Computing and Informatics* 17.1 (2021): 168-192. doi: 10.1016/j.aci.2018.08.003.
- [15] R. Iyengar, Scaling Up Wearable Cognitive Assistance for Assembly Tasks, PhD's thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2023. UMI order number: CMU-CS-23-112. doi: 10.1184/R1/23302121.v1.
- [16] D. Zubov, A. Aljarbouh, A. Kupin, and N. Shaidullaev, Spatial Cognition by the Visually Impaired: Image Processing with SIFT/BRISK-like Detector and Two-keypoint Descriptor on Android CameraX, in: A. Nandal, L. Zhou, A. Dhaka, T. Ganchev, F. Nait-Abdesselam (Eds.), *Machine Learning in Medical Imaging and Computer Vision*, IET, Stevenage, UK, 2023, pp. 249-276. doi: 10.1049/PBHE049E_ch12.
- [17] Mehak Maqbool Memon, Manzoor Ahmed Hashmani, Aisha Zahid Junejo, Syed Sajjad Rizvi, and Adnan Ashraf Arain. "A Novel Luminance-Based Algorithm for Classification of Semi-Dark Images." *Journal of Applied Sciences* 11.18 (2021). doi: 10.3390/app11188694.
- [18] Yantong Chen, Wei Xu, and Yongjie Piao. "Target Matching Recognition for Satellite Image based on the Improved FREAK Algorithm." *Mathematical Problems in Engineering*, 2016 (2016). doi: 10.1155/2016/1848471.
- [19] Wikipedia, Road signs in Kyrgyzstan, 2023. URL: https://en.wikipedia.org/wiki/Road_signs_in_Kyrgyzstan.
- [20] D.G. Lowe. "Distinctive Image Features from Scale-invariant Keypoints." *International Journal of Computer Vision* 60.2 (2004): 91-110. doi: 10.1023/B:VISI.0000029664.99615.94.
- [21] Youwei Wang, Lizhou Feng, Jianming Zhu, Yang Li, and Fu Chen. "Improved AdaBoost Algorithm Using Misclassified Samples Oriented Feature Selection and Weighted Non-negative Matrix Factorization." *Neurocomputing* 508 (2022): 153-169. doi: 10.1016/j.neucom.2022.08.015.
- [22] Pengbo Xiong, Shaokai Wang, Weibo Wang, Qixin Ye, and Shujiao Ye. "Model-Independent Lens Distortion Correction Based on Sub-Pixel Phase Encoding." *Sensors Journal* 21.22 (2021). doi: 10.3390/s21227465.