19th Computer Vision Winter Workshop
Zuzana Kúkelová and Jan Heller (eds.)
Křtiny, Czech Republic, February 3–5, 2014

Evaluation of performance of smart mobile devices in machine vision tasks

Danijel Skočaj, Domen Tabernik, Domen Rački, Matjaž Hegedič, Alen Vrečko, and Matej Kristan

University of Ljubljana, Faculty of Computer and Information Science
{danijel.skocaj,domen.tabernik,alen.vrecko,matej.kristan}@fri.uni-lj.si

Abstract *The recent advance in mobile processing power and imaging devices opened the door to a wide range of mobile vision applications. However, in contrast to typical specially designed machine vision systems in industrial environment, the mobile devices, expected to address the same mobile vision problems, significantly vary in imaging sensors as well as processing power. In this paper we present the results of the experimental study in which we evaluated 22 smart mobile devices in terms of accuracy that can be achieved when using these devices for measuring distances between points in the plane. The results show that smart phones and tablet computers can be used as a high precision measuring device, achieving sub-milimeter accuracy for measurement in the plane defined by a commonly accessible reference object such as an A4 sheet of paper.*

1 Introduction

In recent years we have witnessed a huge growth in popularity of mobile devices such as smart phones and tablet computers. As is often the case, this popularity growth comes hand in hand with the constant growth in performance of the mobile devices in terms of size, processing speed, interaction with the user, integrated sensors, etc. Among sensors, the integrated camera is one of the most important and prominent ones. Nowadays, owning a decently powerful computer and a decent camera, encapsulated in a small mobile device, is often considered a basic social necessity. It seems only natural that these devices are becoming an important platform for application of computer vision and machine vision solutions. These range from devices to aid visually-impaired [6], scene reconstruction [8], augmented reality [11], mobile text translation [9] to visual landmark identification [1]. It is to be expected even a greater use of mobile vision applications in the future.

The history of machine vision has seen many successful industrial applications; part of this success, however, can be certainly attributed to the controllable environment they are usually operating in. When designing a classical machine vision system, such as quality control or a system for optical measurement, there are always three major factors we have to consider: (i) the imaging device we are using, (ii) the environment (especially the illumination of the object and scene), and (iii) the position and the orientation of the object with respect to the scene and the camera. Based on these three factors we can optimize our selection of software and hardware for a specific task (e.g. we can select the camera and the lenses that suit to the problem most). Sometimes, we can (or have to) even adapt the original environmental conditions to meet the requirements of a specific solution. When some of these factors can not be controlled enough, we have to compensate that by robustifying the methods applied.

In contrast, when designing machine vision applications for mobile devices, we have very little control over the factors mentioned above, since it is the global market that dictates the choice of software and hardware (camera included) our application has to run on. All we can do is to state the minimal requirements, which again have to be often as minimal as possible. It can be quite a shock to replace a dedicated industrial camera with a plethora of different mobile phone cameras, primarily designed for social networking. Clearly, in this case the robustness of our machine vision methods faces much greater challenges. In addition our applications are now supposed to be used by non-expert users, and must, hence, be easy-to-use and possibly conform to other related requirements.

In this paper we discuss some of the challenges we face when implementing machine vision applications on smart mobile devices. Our focus is on applications that require very accurate image formation and detection, as well as localization of the individual parts of the image (e. g. for accurately measuring the size of the captured objects). We conducted an experimental study in which we assessed the ability of different models of mobile devices to handle such tasks.

Several authors have studied computer vision algorithms in the context of the processing power of modern mobile devices. Recently [2] have published results of a large-scale experiment that benchmarks computer vision algorithms potentially interesting for mobile applications. They analyze single and multi-thread implementations of these algorithms and identify major areas where architecture can improve the performance. Wang et al. [10] consider the possibility of levereging the GPU in mobile phones to speed up computer vision algorithms. Our focus was on the analysis of imaging capabilities of these devices in the context of machine vision tasks. In total we evaluated 22 mobile devices. We present the results on their appropriateness for using them as vision-based measuring devices.

The paper is organized as follows. In Section 2 we present the methods that were used to evaluate the smart mobile devices. In Section 3 we describe the evaluation proto-

col and then present the evaluation results in Section 4. We conclude with the final remarks and the future work.

2 Methods

2.1 Problem description

We can instantiate the general questions stated in the introduction with the following specific problems: How accurately can we measure distances between points in a plane with smart mobile devices and how do different mobile devices differ in their performance? Since no additional information about various devices is known in realistic scenarios, we are restricted to the information that can be obtained directly from the device and the captured image. No explicit calibration of the camera is therefore allowed.

Since we want to measure the distances in the metric absolute space, we need the means of determining the scale of the elements in the captured image. In principle, we could obtain that information from the integrated accelerometers and gyroscopes, however the results obtained in this way are not very accurate [7]. Therefore we allow the use of a reference object of a known size to determine the scale. We also assume the widespread use of our application, which means not only that this object is supposed to be of a standard size, but also be very ordinary an easily accessible. Considering all this an A4 sheet of paper seems like a natural choice. Therefore all the measurements that we will make will be done with respect to the A4 sheet of paper. The application has first to automatically detect the edges and the corners of the sheet of the paper from which the camera external parameters can be estimated using standard solutions from camera geometry.

In the following subsections we briefly describe the methods that are used to implement this process. We used the fundamental and well established computer vision techniques. The primary purpose of this implementation was to evaluate a wide range of mobile devices and to empirically estimate the upper bound of the error, and not to advance the individual methods used (which could be done and would slightly improve the overall results).

2.2 Detection of reference object

The procedure for robust detection of the paper sheet edges consists of the following well-known computer vision steps: We first apply the *Canny edge detection algorithm* to the grayscale image containing the reference object. Next, we detect the lines in the binary edge image using the *Hough transform*. Lines that lie within each others' delta regions are grouped together. Lines that withstand the filtering process are then fitted to the points in the binary edge image, using the least squares method. To ensure robust and precise line fitting, we discard the points lying in delta regions of other lines. The remaining set of points is then sorted according to the point-to-line distances. Finally, the lines are fitted to a fraction of their closest points, thus eliminating outliers from the line fitting process. In this way we detect the four edges of the reference object and then calculate their intersections with subpixel accuracy. The four intersection points are then subjects of further processing.

2.3 Estimation of camera parameters

The origin of the world coordinate system is placed in the top-left corner of the reference object, with X and Y axes running along its shorter and longer edge, respectively. To determine the camera orientation, we apply a flat-marker approach often used in augmented reality applications. The *pinhole camera model* [3] establishes the following relation between the 3D paper-sheet corners X_i and their 2D image projections x_i:

$$\mathbf{x}_i = \mathbf{K}[\mathbf{R}|\mathbf{t}]\mathbf{X}_i, \tag{1}$$

where $\mathbf{K}$, $\mathbf{R}$ and $\mathbf{t}$ are the camera calibration, rotation and translation matrices, respectively. The calibration matrix can be estimated directly, by reading the relevant parameters from the mobile device. We assume that the camera sensor matrix is not skewed and that the principal point is well approximated by the image center. The only unknown parameters are then the focal lengths in pixels (f_x,f_y) which can be estimated indirectly from the x and y field-of-view angles α_x and α_y, respectively,

$$f_x = \frac{1}{2}W\tan^{-1}(\frac{\alpha_x}{2})\ ;\ f_y = \frac{1}{2}H\tan^{-1}(\frac{\alpha_x}{2}), \tag{2}$$

where W and H are sensor width and height, respectively, in pixels. These parameters fully define the camera calibration matrix.

With the calibration matrix estimated, we use the standard approach by noting that the 3D corners X_i of the paper-sheet lie on a plane with Z coordinates equal to zero. This means that the transformation of 3D points on the sheet of paper and their projected image points is governed by a homography matrix $\mathbf{H}$, i.e.,

$$\mathbf{H} = \mathbf{K}[\mathbf{r}_1, \mathbf{r}_2, \mathbf{t}], \tag{3}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the first two columns of the rotation matrix. The homography is calculated directly from the 2D-3D correspondences. Since $\mathbf{H}$ and $\mathbf{K}$ are known, we can calculate $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{t}$ by inversion of (3) and from the orthogonality constraint we get the last column of the rotation matrix as $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$.

3 Experimental protocol

3.1 Calibration

In our experimental study we used a reference object with a printed checkerboard pattern with a known size of the squares. Similar calibration patterns are commonly used in computer vision for camera calibration. In fact, as a first step of our study we calibrated the camera of every device using this pattern and estimated the intrinsic and extrinsic camera parameters. Calibration was performed using the Camera Calibration Toolbox for Matlab[1]. Since the calibration was done in a controlled environment we considered these parameters as as accurate as possible and used them as a reference.

[1] http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

3.2 Test application

To standardize the process of capturing evaluation images, we developed a special Android application that guides the user through the process of image acquisition (see Fig. 1). Besides ensuring the uniformity of image acquisition, this approach also simulates potential real mobile applications. A blue wireframe that was supposed to be approximately aligned with the paper sheet edges guided the user how to position the camera. 18 images were captured from the hemisphere above the calibration pattern. All of these images were used in the processes of camera calibration and evaluation.

In addition, we implemented several standard computer vision algorithms that were run on every Android mobile device to test the CPU performance. OpenCV[2] implementation of these algorithms was used.

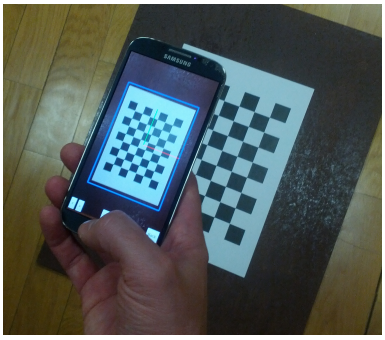


Figure 1: Test application.

3.3 Evaluation procedure

After the evaluation images were captured, we applied the procedures described in Section 2 to calculate the homography, that mapped the image points into the world coordinate frame defined by the reference object. The following estimates and errors were then calculated:

- **Error in corner detection.** Corner points of the individual squares that were detected in the process of calibration were mapped to the world coordinate frame using the estimated homography. We calculated the distance (in mm) between the detected corner points and the ground truth. In this way we estimated the error that was caused by the approximative estimation of homography, reflecting the inaccuracies in image formation and reference object corner points detection.
- **Error of estimated checkerboard width.** We calculated the distance between two specific points in the image, i.e. the width of the checkerboard (the distance between the first and the last column of squares) and compared it to the ground truth.

We also wanted to estimate the influence of the radial distortion in the images. We therefore calculated the above mentioned errors in three ways, i.e. by processing three types of images:

- **Original images**. We used the original captured images without any preprocessing; no image undistortion was considered.
- **Images undistorted with specific distortion coefficients**. We processed the original images, however we corrected all the detected image points (before we mapped them into the metric world coordinate frame) using the estimated camera matrix and distortion coefficients obtained in the process of the calibration of the same camera that was used to capture the image. We therefore achieved a similar result as it would be achieved by processing images undistorted with the specific distortion coefficients.
- **Images undistorted with generic distortion coefficients**. In general use, the specific distortion coefficients are not known. We therefore also performed the evaluation using generic distortion coefficients (obtained as a median of the distortion coefficients of all evaluated cameras). These parameters were fixed and used with all evaluated devices.

We also evaluated the obtained camera parameters:

- **Intrinsic parameters**. We estimated the intrinsic camera parameters and compared them with the parameters obtained in the calibration process.
- **Camera position**. By using the estimated intrinsic camera parameters and the estimated homography we estimated the positions of cameras during the image acquisition with respect to the reference object. We compared these positions with the extrinsic camera parameters obtained in the calibration process.

Finally, we measured the processing power of the individual devices. We therefore measured the **performance of the CPUs**. On every Android device we run four tests composed of the algorithms commonly used in computer vision:

- *Canny*. Canny edge detection.
- *Hough*. Line detection with Hough transform.
- *GrabCut*. Image segmentation with GrabCut method.
- *Gauss*. Image smoothing with Gaussian kernel.

Every test was executed on different images with different resolutions. The same images and algorithms were used on all evaluated devices.

4 Evaluation results

In this section we present the results of the experimental evaluation. We tried to estimate the errors up to 0.1 mm accuracy. However, since we did not have an adequate highly accurate measuring device to measure the ground truth data on the reference object, we probably also made small measurement errors, which, however, did not significantly influence the overall results and the main conclusions we can draw from them.

[2] http://docs.opencv.org

There was also a small number of images (around 3%), were the homography estimation process described in Section 2 failed due to several reasons, mainly due to bad quality of the captured images. Since the robustness of the homography estimation process was not a topic of this research, the results obtained on such images were not considered in this experimental study.

4.1 Evaluated devices

We evaluated 22 smart mobile devices; 12 Android phones, 6 Android tablets, and 4 iOS devices. The list of the devices is presented in Table 1.

Table 1: Evaluated smart mobile devices, CPU type, version of operating system.

No	Device	CPU	OS
1	Samsung Nexus	v7a	4.1
2	Samsung Galaxy Note 2	v7a	4.1
3	Samsung Galaxy S4	v7a	4.2
4	Sony Xperia	v7a	2.3
5	Huawei U8850	v7a	2.3
6	HTC Desire HD A9191	v7a	2.3
7	Samsung Galaxy S3 mini	v7a	4.1
8	Samsung Galaxy S2 GT-I9100	v7a	4.1
9	Sony Xperia ST27i	v7a	2.3
10	HTC Sensation	v7a	4.0
11	HTC Wildfire	v6j	2.3
12	Samsung Galaxy Mini GT-S5570	v6	2.3
13	Samsung Galaxy Tab 10.1 GT-P7500	v7a	3.2
14	Asus TF300T	v7a	4.2
15	Samsung Galaxy Tab 10.1 GT-P7500	v7a	3.1
16	ASUS Transformer TF201	v7a	4.1
17	Samsung Galaxy Note 10	v7a	4.1
18	Samsung Galaxy Tab 2 7.0 GT-P3110	v7a	4.1
19	Apple iPhone 4	A4	6.1
20	Apple iPhone4S	A5	6.1
21	Apple iPhone 5	A6	6.0
22	Apple iPad 2	A5	6.1

4.2 Checkerboard detection

4.2.1 Error in corner detection. In the first experiment we compared the locations of the detected corners of the checkerboard pattern (in mm) with the reference (known) coordinates. The results obtained by processing the original images are presented in Fig. 2.

The average error was 0.34 mm (with st. deviation 0.35 mm) and the median error was 0.28 mm. The system was therefore able to very accurately detect the corners of the A4 paper sheet and to correctly estimate the homography. The top-left plot in Fig. 2 presents the error histogram; most of the errors are smaller than 0.5 mm, almost all of them are smaller than 1 mm.

The top-right plot presents the errors obtained with the individual devices; the device numbers correspond to the numbers listed in Table 1. With the exception of a couple of devices, the results do not differ to a large extent. One has to note that we have to be rather careful when interpreting these results. Although the images were taken from the approximately same directions on all devices, they were taken by different persons, with different vigilance in different illumination conditions, therefore a direct comparison between the results of different devices is not very appropriate.

The bottom-left plot presents the average error by image numbers (i.e., viewing angles) while the bottom-right plot depicts the average error for all 48 points (corners) on the checkerboard (listed from top left to bottom right). The latter plot indicates that the error in the middle of the checkerboard is smaller than the error at its edges, which could be caused by a different influence of the radial distortion.

We wanted to better check the role of the radial distortion, therefore we undistorted the images (i.e., the detected points) before estimating the homography and projecting the detected points in the world coordinate frame as described in Section 3. The results are shown in Fig. 3(a). The bars depict the mean error for all three types of input data: original images (*orig*), points corrected using specific (*undSpec*) and generic distortion coefficients (*undGen*). The red lines depict the error median. The first plot shows the mean error in localisation of pattern corners. The mean error slightly reduces when the distortion coefficients are considered.

Fig. 4 shows more detailed results obtained with the generic distortion coefficients. By comparing this figure with Fig. 2 we can see that the results improve for most of the devices. There are a few exceptions, however; the devices where the distortion coefficients deviate from the general trend, and they should be treated separately.

4.2.2 Error of estimated checkerboard width. The errors in estimating the checkerboard width are presented in Fig. 3(b). In this task the correction of the radial distortion proved to be very useful; the mean width error decreased from the 0.91 mm to 0.55 mm. This is probably due to the fact that two points between which we measured the distance lay on the edge of the reference object where the influence of the radial distortion is bigger. A very similar improvement of the results we obtained in both cases; when the specific as well when the generic distortion coefficients were used. This is rather a positive finding, since this means that the cameras on mobile devices are sufficiently similar that can be modelled using the same set of distortion coefficients.

4.3 Camera parameters

4.3.1 Image resolution Fig. 5 depicts image sizes that were used when capturing images with different devices. We can see that the resolution (width and height of images) considerably vary across devices. This is partially due to different resolution of the camera sensors. Additionally, we did not always use the maximal camera sensor size due to limitations of certain devices, where the ratio of the width versus the height must match with the selected camera preview size and with the actual captured picture size, otherwise the device would produce invalid picture or it would not produce the same picture as seen on the display (preview may have been cropped). Being restricted by those limitations in some cases we selected smaller picture sizes than maximal

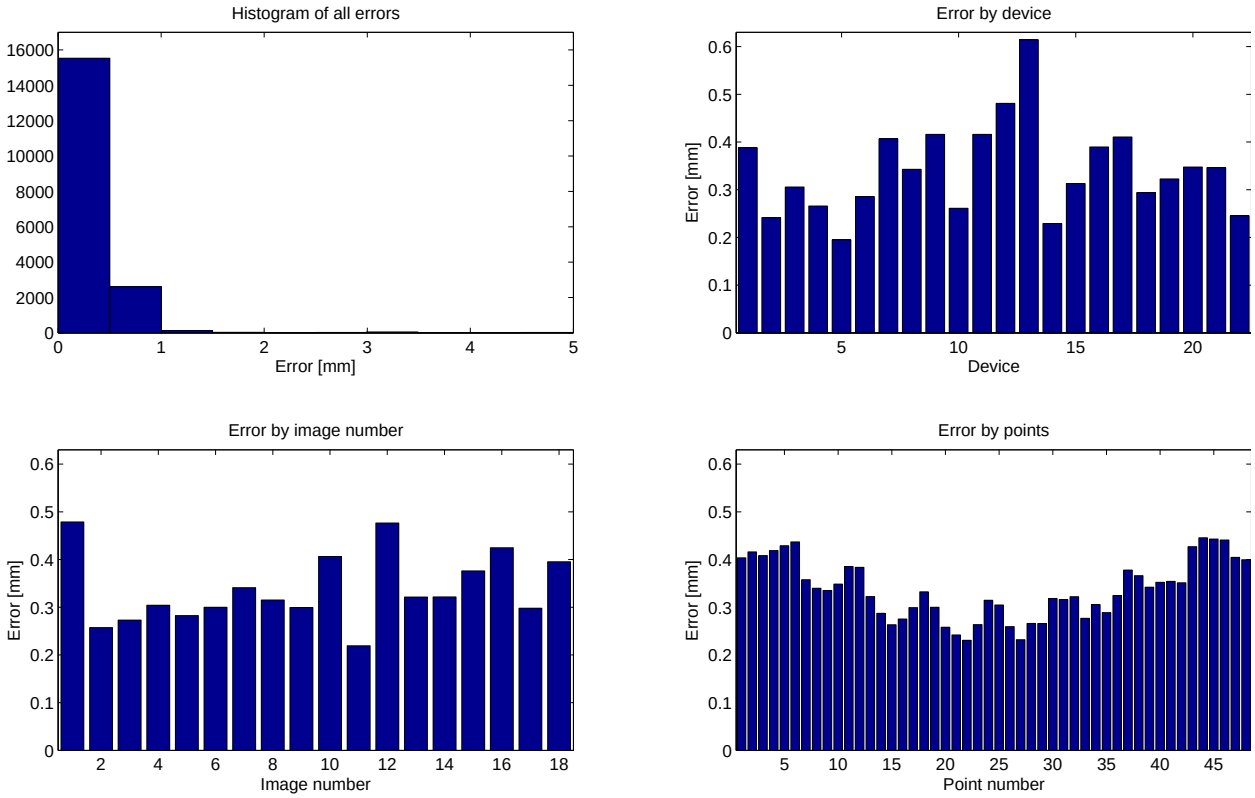
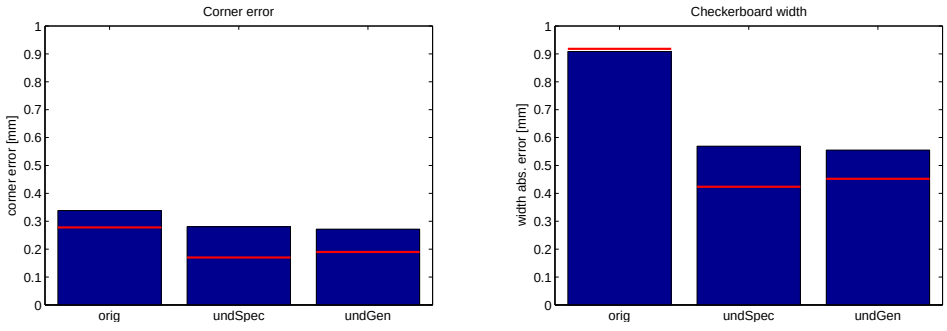


Figure 2: Results obtained on original images: histogram of all errors, error by device, error by image number and error by points.



(a) (b)

Figure 3: Results obtained by processing different types of images (i.e., points): (a) corner error, (b) checkerboard width error.

supported by the camera.

4.3.2 Intrinsic camera parameters. In this experiment we tried to estimate the intrinsic camera parameters from the information provided by the mobile device (as described in Section 2) and compare these parameters with the reference values obtained in the calibration process.

Fig. 6 depicts the results for the individual devices. The top-left plot compares the calibrated and the estimated focal lengths. We can see that the estimates are mostly sufficiently close to the reference values. A similar observation we can also make about the top-right plot, which shows the coordinates of the principal points.

The bottom-left plot shows the pixel "squareness", i.e. the ratio between the width and the height of the individual imaging element. Our assumption (that the elements are square) is mostly confirmed with the parameters obtained in the calibration process. We can see that squareness is almost always very close to 1; small deviations are probably caused by the imperfect calibration.

The bottom-right plot depicts the calculated ratios between the width and the height of the imaging sensor obtained by considering viewing angles as reported by the imaging device as well as the ratios obtained by considering the maximal resolution of the images as offered by the camera. We can see that at certain devices there is a considerable deviation, mainly due to not sufficiently accurate specification of the viewing angles.

Nevertheless, we can conclude that only by considering the data provided by the imaging device we can construct a reasonably credible camera matrix. However, it would be worth comparing the estimated focal length with the focal length obtained directly from the images by, e.g., considering the detected known four or five points in the reference object [5, 4].

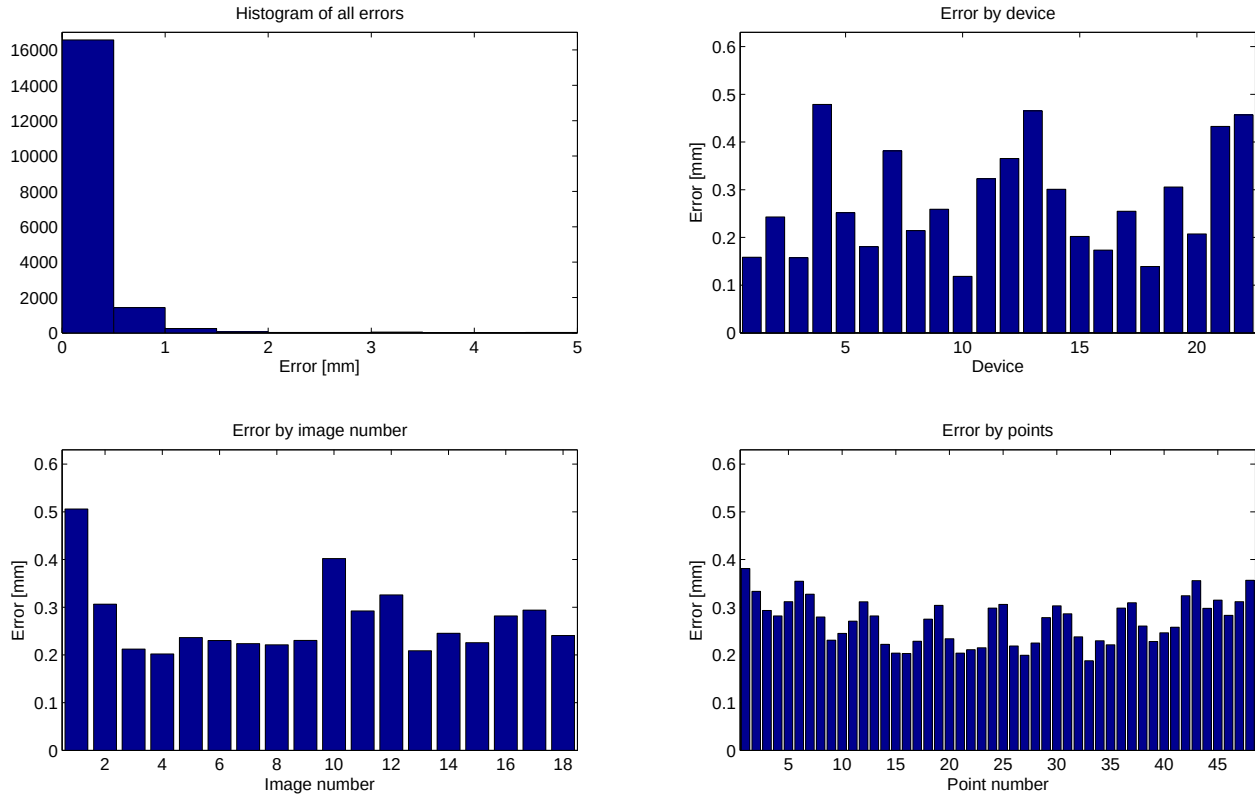


Figure 4: Results obtained on images undistorted with generic distortion coefficients: histogram of all errors, error by device, error by image number and error by points.

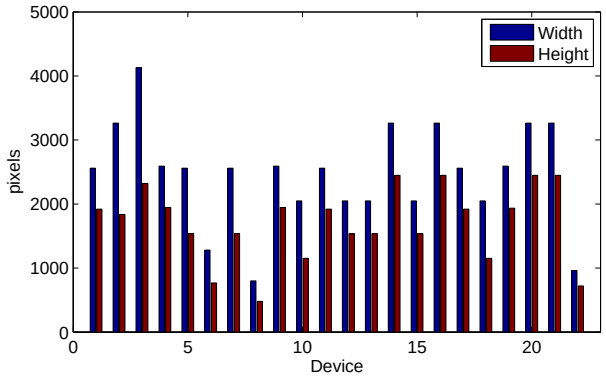


Figure 5: Image resolution that was used when capturing images.

4.3.3 Camera position In this experiment we compared the estimated camera positions (obtained as explained in Section 2) with the camera extrinsic parameters obtained in the calibration process. The overall mean results are presented in Fig. 7(a). The mean euclidian distance between the estimated and calibrated camera positions are presented, as well as the distances along the x, y, and z axes. We can see that the error is relatively small, since it does not exceed 1 cm along individual axis. Fig. 7(b) plots mean errors for every device. Except a few exceptions, there are no large deviations.

4.4 CPU performance

To conclude the experimental section, we present the results of CPU performance evaluation on the Androind smart mobile devices presented in Table 1. Rather than relying on many publicly available performance comparisons of mobile devices, we decided to do our own performance evaluation, using typical computer vision algorithms. Hence we developed a test application where we implemented the four standard computer vision algorithms as described in Section 3.3.

We measured processing times for each evaluation image and calculated the weighted means for all four algorithms. The weight was determined based on the size of the corresponding image. The results are expressed in terms of milliseconds necessary to process 1000 pixels. They are presented in Figure 8. As the processing time depends on the content of the image, the absolute values are not very informative. However, the relative comparison of these results indicates well the differences in the performance of the individual devices. As we can see two devices (no. 11 and 12) stand out from the rest. We can see from Table 1 that these are the only two devices that do not feature the *armeabi-v7a* architecture, but an older one. The other platforms achieved comparable results: a few exceptions apart, they were less than 2.5 times slower than the fastest. Based on these evaluation results we can deem the *armeabi-v7a* mobile architecture suitable for running relatively demanding machine vision application. The version of the Android operating system had no influence on the evaluation results.

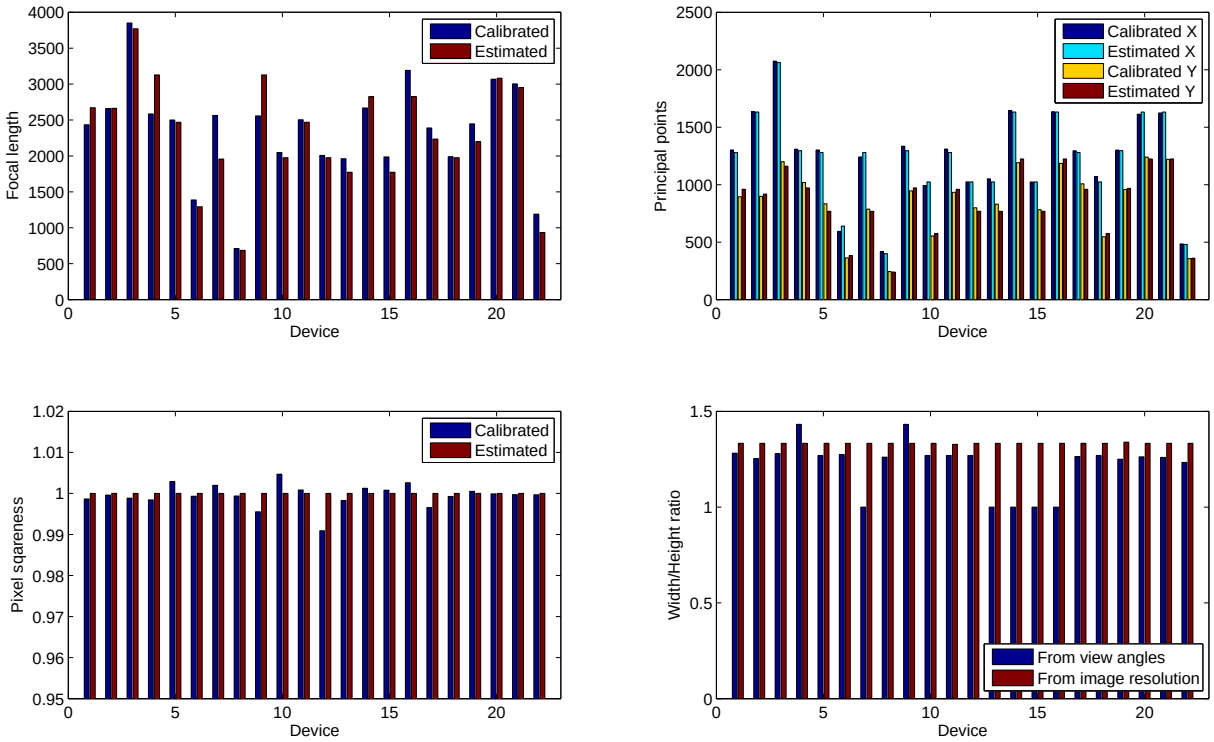


Figure 6: Intrinsic camera parameters. Comparison between parameters obtained with calibration and estimated from the factory settings. (a) Focal length in pixels. (b) Principal point. (c) Pixel squareness. (d) The ratio between the image width and height.

5 Conclusion

In this paper we have explored the feasibility of using a modern mobile phone or a tablet as an integrated device for performing visual measurements. We have constrained our scope of applications to flat-marker-based systems. Such a system uses a known reference object to estimate the camera parameters. To maximize the applicability we have chosen a widely-available reference object – the A4 paper sheet. We have developed a simple processing pipeline that detects the sheet of the paper. Using known dimensions of the sheet, we estimate the external parameters, while the camera internal parameters are constructed by reading the parameters provided by the device.

We have performed an in-depth analysis of accuracy of our mobile measuring system, using 22 different mobile devices. The results show that the error in detecting the corners of the checkerboard pattern is below 0.5 mm in all devices. Considering the size of the reference object, the relative error is below 0.2%. We have observed that the radial distortion is indeed significant when considering sub-milimeter measurements. A notable result is that generic distortion parameters (obtained from calibration results over all devices) reduce the errors induced by the radial distortion. This implies similar lens across the different phones. The analysis of approximation of internal parameters showed that the focal length is sufficiently well approximated by reading the viewing angle directly from the device, as well as the principal point is well approximated by the image center. The analysis of the extrinsic parameters estimation shows that homography-based approach introduces errors that are below 10 mm in all three coordinate axes, making the overall camera position error smaller than 15 mm. We have further measured the variability of the processing capabilities of the tested mobile devices by running some basic computer vision operations. Results show that the processing power does not prohibitively vary in this respect for the newer architectures, however, the processing is significantly slower for the older ones.

Our results show that modern smart phones and tablets can indeed be used as a high-precision measuring device even using a simple flat-caliber-based approach, achieving sub-milimeter accuracy in measurements in the reference object plane. An important result is also the fact that intrinsic camera parameters, along with radial distortion coefficients, may be estimated without specialized calibration procedures. This is especially important property as it significantly widens the application domain of mobile-phone-based measuring systems.

Our future work will extend in several directions. We will study accuracy of more advanced approaches for camera parameters estimation (e.g., [5, 4]) in our setting. Another direction of further research will be to conduct the feasibility study of high-precision measuring of 3D objects placed in the center of the sheet of paper.

References

[1] D.M. Chen, G. Baatz, K. Koser, S.S. Tsai, R. Vedantham, T. Pylvanainen, K. Roimela, Xin Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk. City-scale landmark identification on

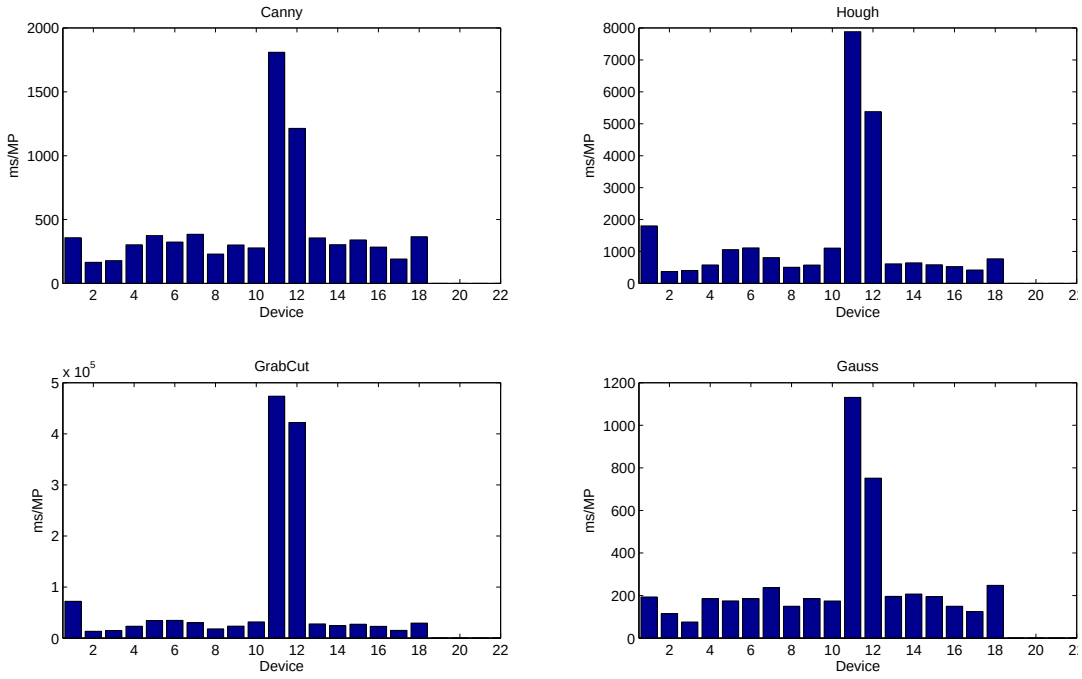


Figure 8: CPU benchmark results.

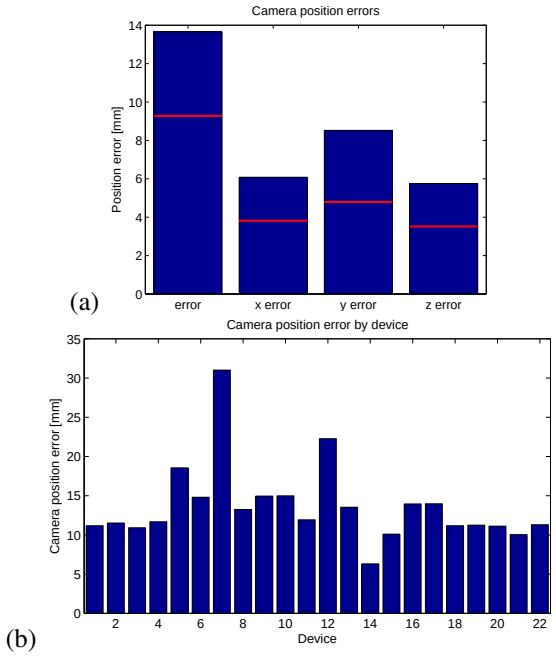


Figure 7: Camera position errors with respect to the positions obtained in the calibration process: (a) overall error, (b) errors of individual devices.

mobile devices. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 737–744, 2011.

[2] J. Clemons, Haishan Zhu, S. Savarese, and T. Austin. MEVBench: A mobile computer vision benchmarking suite. In *Workload Characterization (IISWC), 2011 IEEE International Symposium on*, pages 91–102, 2011.

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[4] Z. Kukelova, M. Bujnak, and T. Pajdla. Real-time solution to the absolute pose problem with unknown radial distortion and focal length. In *International Conf. Comp. Vision*, 2013.

[5] Z. Kukelova, J. Heller, and T. Pajdla. Hand-eye calibration without hand orientation measurement using minimal solution. In *Computer Vision – ACCV 2012*, volume 7727 of *Lecture Notes in Computer Science*, pages 576–589, 2013.

[6] R. Manduchi, S. Kurniawan, and H. Bagherinia. Blind guidance using mobile computer vision: A usability study. In *Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '10, pages 241–242, 2010.

[7] L. Meier F. Camposeco Paulsen O. Saurer M. Pollefeys P. Tanskanen, K. Kolev. Live metric 3D reconstruction on mobile phones. In *Proc. of Int. Conf. on Computer Vision, ICCV 2013*, Sydney, 2013.

[8] Q. Pan, C. Arth, G. Reitmayr, E. Rosten, and T. Drummond. Rapid scene reconstruction on mobile phones from panoramic images. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 55–64, 2011.

[9] M. Petter, V. Fragoso, M. Turk, and Charles Baur. Automatic text detection for mobile augmented reality translation. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 48–55, 2011.

[10] Guohui W., Yingen X., J. Yun, and J.R. Cavallaro. Accelerating computer vision algorithms using OpenCL framework on the mobile GPU - a case study. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 2629–2633, 2013.

[11] Y. Xin and C. Kwang-Ting. LDB: An ultra fast feature for scalable augmented reality on mobile devices. In *Mixed and Augmented Reality (ISMAR)*, pages 49–57, 2012.