BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 63, No. 4, 2015
*DOI: 10.1515/bpasts-2015-0102*

# LEDs based video camera pose estimation

K. SUDARS*, R. CACURS, I. HOMJAKOVS, and J. JUDVAITIS

Institute of Electronics and Computer Science, 14 Dzerbenes St., LV-1006, Riga, Latvia

**Abstract.** For 3D object localization and tracking with multiple cameras the camera poses have to be known within a high precision. The paper evaluates camera pose estimation via a fundamental matrix and via the known object in environment of multiple static cameras. A special feature point extraction technique based on LED (Light Emitting Diodes) point detection and matching has been developed for this purpose. LED point detection has been solved searching local maximums in images and LED point matching has been solved involving patterned time functions for each light source. Emitting LEDs have been used as sources of known reference points instead of typically used feature point extractors like ORB, SIFT, SURF etc. In such a way the robustness of pose estimation has been obtained. Camera pose estimation is significant for object localization using the networks with multiple cameras which are going to an play increasingly important role in modern Smart Cities environments.

**Key words:** camera pose estimation, image keypoint detection and matching, 3D point reconstruction, object localization and tracking.

## 1. Introduction

Object localization using multiple cameras is still a widely studied area in computer vision. Much progress has been made to improve SLAM (Simultaneous Localization and Mapping) and there are many developed approaches and papers published solving SLAM problems, 3D reconstruction from images, object localization via cameras (etc.) [1]. The correct object localization using multiple cameras requires knowing of camera poses with high precision. It would allow object localization via triangulation without solving complicated optimization equations in real time.

Similar camera pose estimation techniques based on LED detection can be found in literature. In [2] and [3] works are very similar. In both papers the same LED detection technique is used where LEDs are lighted in sequence and LEDs are synchronised with camera so that it is known what LED was on for each captured image. The LED detection is based by taking the brightest pixel value, and further improving coordinate accuracy by sub-pixel analysis phase by taking weighted average of pixels neighborhood. This work does not consider robustness to cases where other bright objects appear in the scene. In contrast out method accounts for cases where other bright objects could be in scene.

In [4] infrared LED's are used in known position to compute camera pose. The usage of infrared LEDs allow simple LED detection using basic thresholding technique, but requires usage of cameras with infrared-pass filter. In smart city environments this requirement can limit the number of usable cameras in existing infrastructure. Also their LED detection method does not distinguish LEDs and to find LED correspondences camera pose is computed for every possible combination by taking some at minimum three LED coordinates and by calculating the reprojection error of other LEDs the combination which gives the least reprojection error is taken to be correct one. This makes this method very slow for larger number of LEDs which can limit accuracy of method, where increasing number of LEDs can increase accuracy. In contrast in out method each LED has different blink pattern making each lead distinguishable.

In more detail the paper evaluates two approaches of multiple camera pose estimation: camera pose estimation via the fundamental matrix and pose estimation via a known object. Feature points extracted from LED sources are used as input data for both methods in case of pose estimation. The special feature point extraction technique based on LED point detection and matching has been engineered for this purpose. LED point detection has been solved searching for image local maximums and LED matching has been solved involving patterned time functions for each light source. Using the LED approach the robustness of pose estimation could be achieved and the method will not suffer from insufficient number and quality of point matches as it could happen using feature extraction based only on image processing. On the other hand, camera pose estimation using feature extractors like ORB, SIFT, SURF (etc.) of course has their own advantages and it is convenient because all calculations could be done remotely processing video streams from the network cameras. Nevertheless the parameter estimation using engineered LED point extractor is suggested and it excludes worst case scenarios when a good fundamental matrix cannot be estimated due to insufficient quality of extracted feature points. Evaluation of camera pose precision is a hard task (because real poses still remain unknown) and in the paper it has been done evaluating object localization and tracking quality. Typically object localization using multiple cameras could be divided into following steps [1]:

1. Feature point extraction and matching;
2. Camera pose calculation;
3. Object localization (point 3D reconstruction).

*e-mail: sudars@edi.lv

All steps allow variations and the goal of each step could be achieved differently. In the paper (step A) feature points have been extracted and matched using (1) LED detection and matching for camera pose estimations described further and (2) keypoint extractor ORB used for object localization. Step B camera pose calculation has been considered: (1) camera pose estimation decomposing fundamental matrix into rotation and translation, (2) camera pose estimation using known object. In static camera case step B has to be done only once. Step C object localization or object point reconstruction in 3D space from images has been solved using back-projection.

Next sections of the paper are aligned according to these (A, B, C) steps: LED feature point extraction for pose estimation in Sec. 2, camera pose calculation in Sec. 3, object point (obtained using keypoint extractor ORB) localization via back-projection in Sec. 4. Experimental results of point localization in 3D space using estimated camera poses are described in Sec. 5. The made efforts are considered within Smart Cities and its environments where specialized assistant networks via multiple video cameras are able to localize, track and recognize objects.

## 2. Feature point extraction and matching

Calculating camera poses the corresponding points in two paired images have to be known precisely. Taking this into account the initial step is introduced where LED points of a priori defined positions are estimated. For LED point tracking in images a special technique is described further in the paper and evaluated on examples. Estimation of camera poses as a separate step is considered for a reason that typical feature point extractors like ORB, SIFT, SURF might not provide sufficiently good results. For example in cases of blank scenes there could be little or no feature points detected. Also in the scenes with multiple similar feature points, wrong matches between images can be produced, requiring usage of algorithms that are robust to outlier matches like RANSAC. Also feature point geometric coordinates can be determined imprecisely. Usage of ORB, SIFT, SURF etc. is considered after camera calibration triangulating extracted keypoints into 3D space.

**2.1. Technique for LED point extraction from videos.** The technique for camera pose estimation is described further. It is based on observation of local maximum points during some time period (processing video frame sequence over some time). For LED source position extraction in images we have considered the following processing scheme: **Threshold – Select region Local maximum – Correlation – Point selection**.

The special test module with LEDs has been created in order to obtain experimental results. This module includes an array of LEDs controlled by a single-board computer (Raspberry Pi) via GPIO interface as it can be seen in Fig. 2. The program has been written to control switching ON and switching OFF of the LEDs to form various light patterns.

**Threshold.** Firstly, we threshold the image intensity because it is known that LEDs should be located on brightest parts of images. The threshold should separate shining objects from background. In our case we have used 70–95% of maximal image value intensity as a threshold (shown in Fig. 3).

$$I^*(u,v) = \begin{cases} I(u,v), & \text{if } I(u,v) > 0.8 \ \max(I(u,v)) \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $I(u,v)$ – image intensities within one video frame; $I^*(u,v)$ – thresholded image; $u$ and $v$ – image row and column numbers $u = \overline{1,U}$, $v = \overline{1,V}$. Image size $U \times V$.

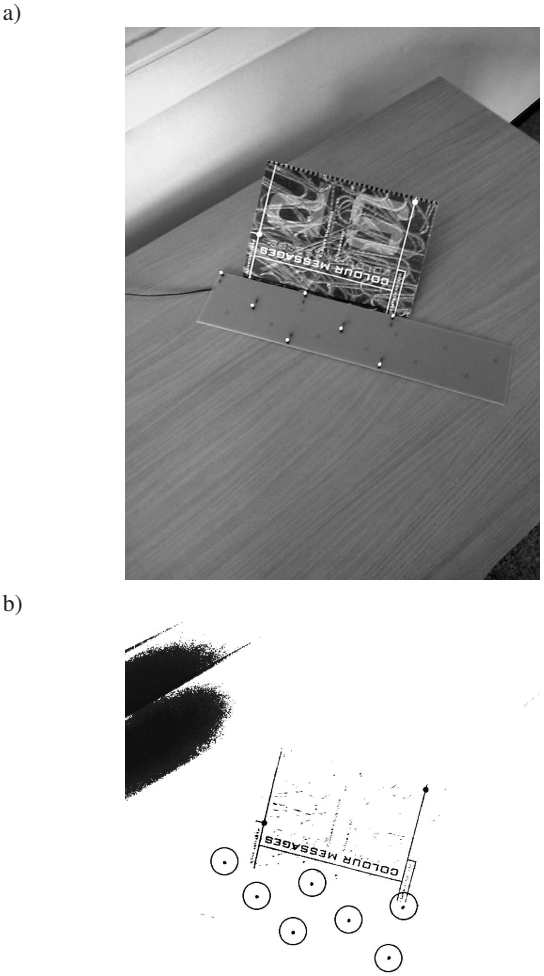



Fig. 1. Image after threshold and LED localization: a) image at beginning, b) thresholded image and located LEDs

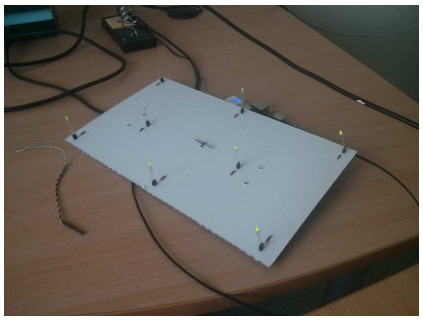


Fig. 2. Input image for LED detection

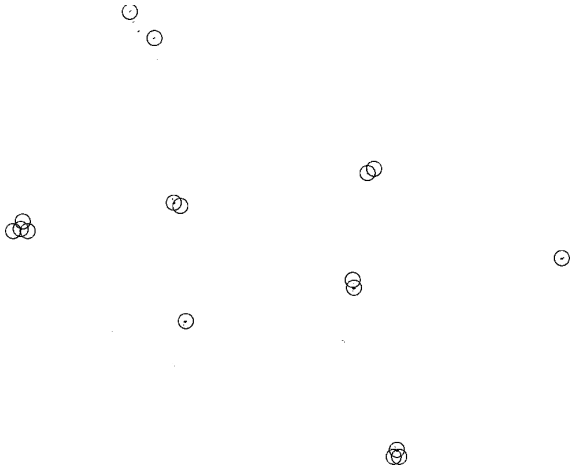


Fig. 3. LED point canditates

**Select regions.** Next thing is to select region of interest. We use various size pixel regions (e.g. $64 \times 64$) as a window in order to select region from image. The region is shifted with a step half of window length (in this case 32). Size of windowing depends on image resolution and the main requirement here is that window size should be large enough to fit LED point and small enough to cut-off other shining and bright regions. Ideally LED point should be alone in a window on a zero background.

$$H_{i,j}(a,b) = I^*\left(\frac{i-1}{2}A + a, \frac{j-1}{2}B + b\right), \tag{2}$$

where $H_{i,j}$ – selected image region; $a, b$ – region element indexes $a = \overline{1,A}$, $b = \overline{1,B}$; $i, j$ – selected region indexes (window number); $A$ and $B$ – window size. After region selection it could be discarded if the following criteria is not met:

$$C_1 < \sum_{a=1}^{A}\sum_{b=1}^{B} H_{i,j}(a,b) < C_2, \tag{3}$$

where $C_1$ and $C_2$ – constants used to discard over-shined regions and noise. These numbers depend on LED brightness and therefore are influenced by distance between cameras and LED points. This criteria also allow to reduce computation burden in further processing. From our experiment it were seen that it is better to choose coefficients which allow to keep all LED points and then find them from many candidates than loose some LED points trying to reduce candidate number.

**Local maximums.** After region selection we take local maximum coordinates as LED point candidate. Of course other criteria could be considered. Nevertheless local maximum is computationally efficient criteria in comparison for example to convolution (taking maximum after convolution). Local maximum values are following:

$$z_{i,j} = \max(H_{i,j}(a,b)), \tag{4}$$

where $z_{i,j}$ – maximum value in particular image region $H_{i,j}$. The coordinates of local maximum are following:

$$p_{i,j} = (l,m) \mid z_{i,j} = I^*(l,m)$$
$$\text{and} \quad (i-1)\frac{A}{2} \le l \le i\frac{A}{2}, \quad (j-1)\frac{B}{2} \le m \le j\frac{B}{2}, \tag{5}$$

where $p_{i,j}$ – image element coordinates corresponding to maximum value in each image region numbers $i$, $j$. Candidate point extraction of considered technique has been evaluated processing 44 test images (available on the server using Link 1: failiem.lv/u/slmwrxr) and the results are summarized in Table 1. The results of processed images are available at Link 2: failiem.lv/u/mxfttlt.

Dependence on range, illumination and other bright objects should be taken into account. The provided results could quickly drop as it is expected into considered videos at Link 3: failiem.lv/u/zqbdjrh. There are other overshined objects too close to LED points and therefore LEDs cannot be separated from other bright regions using windowed local maximums.

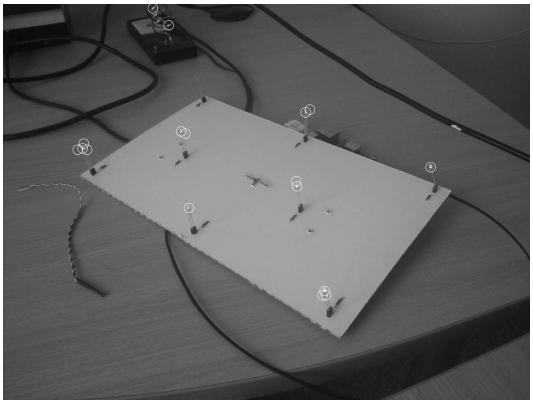


Fig. 4. Selected LED points

Candidate points also could be united by choosing point with larger pixel intensity or averaging if the distance between them is close. It will reduce number of candidates.

**Correlation and point selection.** After candidate point selection it is observed for a time. The candidate is taken as LED point if it emits required pattern of ON-OFF signals during the video frames

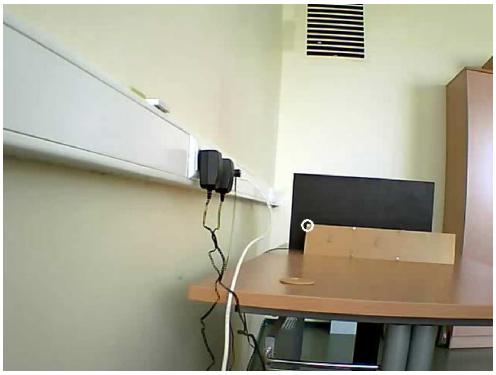
$$r_{i,j}(t) \in \{0,1\}, \tag{6}$$

where $r_{i,j}(t)$ – ON-OFF signal corresponding to candidate point coordinates $p_{i,j}$ in frames $t$.

For matching LED points from candidates many criteria could be used together: matched filtering, Fourier transform on selected frequencies, mean values close to zero. For more reliable performance Hamming codes can be incorporated. In two camera case LED point also is likely to be seen in both cameras at the same time. It could be used for correlation calculation as a criteria.

In our research we have used Fourier coefficients on selected frequencies matched to LED point switching frequencies. This is seen in Fig. 5 where the first LED is located in both images (white circles). Processed videos are available at following Link 4: failiem.lv/u/iihapwg.

a)



b)

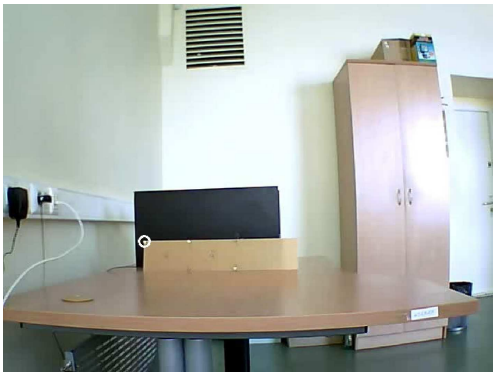


Fig. 5. One LED point matched in both images: a) left image, b) right image

**2.2. Point extraction using ORB.** In order to localize objects we chose to use current state of the art keypoint extraction algorithm from images called ORB (also applicable calculating fundamental matrix). The main paper describing ORB is [5]. Basically ORB consists of FAST (Features from Accelerated Segment Test) [6] corner detector and BRIEF (Binary Robust Independent Elementary Features) [7] descriptor with many modifications targeted to improve the performance. Since ORB is a binary keypoint descriptor it is quite simple to compare two keypoint descriptors by simply calculating their Hamming distance [8]. If two keypoint descriptors from different images produce a Hamming distance smaller than our chosen threshold, then we assume that those two points are at the same location in the real world environment. The main idea behind the ORB is following. FAST corner detector uses exhaustive search on every pixel of the image. Each pixel is checked whether it is a center pixel of a corner. Checking consists of comparing 16 pixels located in a circle (Bresenham circle of radius 3 is used) [6, 9]. Specific center pixel is recognized as corner center pixel if at least 12 contiguous pixels have intensity below or above intensity of center pixel by some set threshold. Such approach allows to greatly optimize each test, because at beginning the only four pixels have to be checked at Bresenham circle locations 1, 9, 5 and 13 to determine whether current point corresponds to the aforementioned statement. FAST corner detector is fast but unfortunately it doesn't contain any information about corners orientation. The corners orientation is estimated using simple and effective corner orientation measurement called intensity centroid [10], which is based on assumption that corners orientation is its intensity deviation from its center and in such a way this vector can be used to describe the orientation of corner. Other part of ORB is BRIEF descriptor. Basically BRIEF is a description of a patch from image by string of bits, which are acquired by performing binary intensity tests. These tests are performed by choosing two pseudo random pixels for each test and comparing their intensity, compare result can be saved in one bit, respectively is the intensity of first pixel above the intensity of second pixel. The implementation of ORB uses 256 such pair of pixels, respectively each point is described by 256 bits. Of course, to compare two different points the same pairs of pixels with respect to the center of point needed to be taken in the same sequence. BRIEF descriptor is fast and it should be adjusted by corner orientation from FAST, in that way, even after in-plane rotation the same pixels are chosen for binary intensity tests. Comparison of most popular descriptors and BRIEF can be found in paper [5]. The ORB algorithm implementation can be found in OpenCV library.

## 3. Camera pose calculation

This section describes camera pose estimation. Two camera configuration is related with their relative rotation and translation. After keypoint matching between two cameras the estimation of rotation and translation could be done. The quality and accuracy of extracted keypoints have direct impact on accuracy of rotation and translation estimation. In the paper the quality of camera pose estimation has been evaluated indirectly via 3D point reconstruction.

**3.1. Calculation of fundamental matrix.** Extended comparison of many fundamental matrix calculation approaches is given in the paper [11]. In practice for good fundamental matrix estimation one object has to be seen in at least two pictures taken from different positions in space. According to rules of perspective geometry each picture or view is in its own coordinate system, where the beginning of coordinate system is view's perspective center. The fundamental matrix ties together two coordinate systems. It is a $3 \times 3$ matrix expressing a transfer and rotation from one coordinate system to another (belonging to one camera and another). It has to be noted that in calculation of fundamental matrix, focal length is normed to be 1. 3D object projection in a 2D plane, is expressed in homogeneous coordinate system [1, 12, 13]

$$p_r^T \mathbf{F} p_l = 0. \tag{7}$$

The fundamental matrix is defined by expression (7), where $p_r$ and $p_l$ is a projection of 3D point expressed in homogeneous coordinate system accordingly in right and left picture. Using expression (7) it is possible to calculate fundamental matrix if a defined amount of corresponding points are found in both pictures. Generally at least 8 matching points are needed, but there is an algorithm which allows calculation of fundamental matrix from only 7 points [1]. In a direct form fundamental matrix can be calculated using 8 matching points

(8 point algorithm). Each match $n$ point $p_l^n = (x_l^n, y_l^n, 1)$ and $p_r^n = (x_r^n, y_r^n, 1)$ together with unknown elements of fundamental matrix constructs the linear equation:

$$x_r^n x_l^n f_{11} + x_r^n y_l^n f_{12} + x_r^n f_{13} + y_r^n x_l^n f_{21} + y_r^n + y_l^n f_{22} + y_r^n f_{23} + x_l^n f_{31} + y_l^n f_{32} + f_{33} = 0. \tag{8}$$

Converting elements of fundamental matrix into form of vectors using row-major, expression (9) can be expressed as a scalar multiplication of vectors:

$$(x_r^n x_l^n, x_r^n y_l^n, x_r^n, y_r^n x_l^n, y_r^n y_l^n, y_r^n, x_l^n, y_l^n, 1)\mathbf{f} = 0. \tag{9}$$

In expression (9) $\mathbf{f}$ is a fundamental matrix in a vector form by row-major layout. Multiple matching points gives a linear equation system, which can be simply described by expression:

$$\mathbf{Af} = 0. \tag{10}$$

In expression (10) matrix $\mathbf{A}$ is called a measurement matrix and it depends on matching point coordinates in both pictures. Each row of this matrix is formed from expression (9). It is obvious that fundamental matrix is $\mathbf{P}$ null space and it can be calculated using linear algebra. For fundamental matrix to have a solution the rank of measurement matrix have to be 8. Some combinations of points leads to unstable results of fundamental matrix calculation, that's why in practice we use more than 8 points. On the other hand if the rank of measurement matrix were higher than 8 then the equations might have no solution therefore in practice singular value decomposition (SVD) is used.

$$\mathbf{A} = \mathbf{U\Sigma V}^T. \tag{11}$$

Using singular value decomposition SVD the measurement matrix is decomposed into three parts where $\mathbf{U}$ is called the left singular vector matrix, $\mathbf{V}$ is called right singular vector matrix and $\mathbf{\Sigma}$ is called diagonal matrix which consists of singular values. The solution of linear equations can be found in the last column on matrix $\mathbf{V}$ [14].

**3.2. Camera pose estimation with fundamental matrix.** For decomposing the fundamental matrix into relative coordinate system's rotation and transition the essential matrix has to be calculated, which is fundamental matrix only with difference that camera focal lengths are normed to camera real values (commonly expressed by pixels) [15]:

$$\mathbf{E} = \mathbf{K}_r^T \mathbf{F} \mathbf{K}_l, \tag{12}$$

where $\mathbf{E}$ – $3 \times 3$ essential matrix; $\mathbf{K}_r$ and $\mathbf{K}_l$ – $3 \times 3$ projection matrices for left and right view. Through SVD we acquire relative translation and rotation of coordinate systems (expressing one view relation to another) [1]:

$$\mathbf{E} = \mathbf{U\Sigma V}^T. \tag{13}$$

SVD (13) produces two solutions for coordinate system relative translation and two for possible rotations. The relative translation of coordinate system $\mathbf{t}$ with $+$ or $-$ sign is the last column of matrix $\mathbf{U}$ from (13). To continue calculation we have to choose the right translation and rotation. It is done by restoring one 3D point using all $\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{t}$ and $-\mathbf{t}$ combinations searching for a solution where reconstructed 3D point is in front of both cameras (reconstructed 3D point $z$ coordinate have to be positive)

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{14}$$

$$\mathbf{R}_1 = \mathbf{UWV}^T, \tag{15}$$

$$\mathbf{R}_2 = \mathbf{UW}^T\mathbf{V}^T. \tag{16}$$

Equations (14), (15) and (16) allow to decompose the fundamental matrix into translation and rotation between two corresponding camera poses.

**3.3. Camera pose calculation via known object.** P$n$P problem (Perspective $n$ point problem) is defined as determining camera pose, when the an intrinsic parameter matrix for camera is known and set of $n$ 3D to 2D point correspondences are determined. Various methods exist for P$n$P problem solving which mainly fit in two groups: non-iterative methods and iterative methods. Non-iterative methods are faster than iterative methods, but in general provide poorer accuracy for camera pose estimation and are less robust to noise from point measurements. For earliest iterative methods, described in: [16–21], time complexity range from $O(n^2)$ to $O(n^8)$, what makes these methods slow for increased number of point correspondences. In contrast one of the latest algorithms, called EP$n$P has time complexity $O(n)$ and provides better accuracy and reduced noise sensitivity than previous methods [22].

The iterative P$n$P solving methods estimate camera pose by iteratively minimizing an appropriate criterion, like reprojection error. Various iterative methods exist in literature: [23–25]. Although iterative methods can achieve excellent accuracy when they converge properly, they are in general slower that non-iterative methods. Iterative estimation methods must be provided with initial guess for camera pose, and estimation speed depends on how close the initial guess is to the actual camera pose. The main weakness for iterative methods are when they are poorly initialized, because then they may fail to converge, or could give solution that gives only local minimum for minimized cost function.

To overcome the drawbacks of iterative methods being slow and non-iterative methods being less accurate, combined methods can be used, where iterative methods are initialized by pose found with non-iterative method. For example in paper [22] it is shown, that EP$n$P algorithm followed by Gauss-Newton optimization gives better accuracy than single EP$n$P method, while computation time increases insignificantly.

For our purposes for camera pose estimation EP$n$P algorithm is used to estimate initial guess of extrinsic camera pose parameters – $\mathbf{P_0}$ for iterative algorithm, which solves camera pose via Levenberg-Marquardt optimisation by minimizing reprojection error. In each iteration camera pose parameters are altered by vector $\boldsymbol{\delta}$ (17)

$$\mathbf{P_i} = \mathbf{P_{i-1}} + \boldsymbol{\delta}. \tag{17}$$

Re-projection error – $\boldsymbol{\epsilon}$ is used as a minimization criterion (18). This error is distance between measured points – $p$ of calibration object in image and re-projected points $f(\mathbf{P_i})$, by projecting those points using computed camera pose (18)

$$\boldsymbol{\epsilon} = d(p, f(P_i)). \tag{18}$$

The update vector $\boldsymbol{\delta}$ can be estimated by solving linear Eq. (19). In this equation $J$ is Jacobian matrix – $\partial f(\mathbf{P_i})/\partial \mathbf{P_i}$

$$(J^T J + \lambda \text{diag}(J^T J))\boldsymbol{\delta} = J^T \boldsymbol{\epsilon}. \tag{19}$$

Parameter $\lambda$ is changed in each iteration. Usually initial value of $\lambda$ is taken around $10^{-3}$. If camera pose parameters in iteration altered by $\boldsymbol{\delta}$ leads to a decreased error, then $\lambda$ is divided by factor 10. Otherwise, if error increases, then $\lambda$ factor is increased by factor 10, until camera pose parameter alteration gives decreased error.

## 4. Object localization in two camera case

After camera position estimation each 3D point could be calculated or localized in 3D space using corresponding image points. It requires to know camera poses and corresponding image points belonging to the object. Corresponding keypoints could be found using one of popular keypoint extractors. Examples are ORB, SIFT, SURF etc.

**4.1. 3D point reconstruction.** The object localization has two parts:

A. Feature point extraction and matching,
B. 3D information reconstruction.

The scheme is the same to camera pose calculation only purpose now is object localization. In our experiments we choose ORB as a good representative of object point extractors (described in Sec. 2). The other keypoint detectors also do not clearly outperform the ORB. The comparison and evaluation of various keypoint detectors could be found [5].

In our paper back-projection is used for 3D point reconstruction from two corresponding image points. At beginning the reprojection matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ have to be constructed [1]:

$$\mathbf{M}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{20}$$

$$\mathbf{M}_2 = [\mathbf{R}\ \ \mathbf{t}], \tag{21}$$

where $\mathbf{R}$ – rotation matrix between two cameras; $\mathbf{t}$ – translation between two cameras.

For each corresponding point pair $n$ the linear equation system can be constructed and solved with SVD:

$$SVD\left(\begin{bmatrix} [p_l^n]_\times \mathbf{M}_1 \\ [p_r^n]_\times \mathbf{M}_2 \end{bmatrix}\right). \tag{22}$$

After singular value decomposition similar to (11) and (13) the homogenized last column of $V$ is the reconstructed 3D point. Of course, other reconstruction techniques exist allowing to cope with corresponding point irregularities caused by imprecise point coordinate estimation etc. These methods have to be considered depending on application.

In our experiments camera pose estimation has been studied indirectly using described 3D point reconstruction.

**4.2. Good keypoint selection with RANSAC.** In case the fundamental matrix is calculated without LED points using some other keypoint detector the additional criteria might have to be required. The common technique is RANSAC for detected point seperation into inliers and outliers [26]. We have used RANSAC to separate matching points from false positives. As a criteria we have used Sampson distance [27] because it is shown in [1] that it gives good results

$$d_s = \frac{(p_r^T \mathbf{F} p_l)^2}{(l_l^1)^2 + (l_l^2)^2 + (l_r^1)^2 + (l_r^2)^2}. \tag{23}$$

In the equation $p_r$ and $p_l$ are two point coordinates in pictures. $l_l$, $l_r$ – vectors of corresponding point epipolar line coefficients in left and right images. It is the corresponding point correlation coefficients of epipolar line on the left image and it is also the corresponding point correlation coefficients of epipolar line on the right image. For each match Sampson distance $d_s$ is calculated and if it reaches threshold $T$ then the match is treated as wrong. In each iteration we count how many matches were found as correct. As the end result we use the iteration which gave the most correct matches.

## 5. Experiment results

The comparison of camera pose estimation via the fundamental matrix and a priori known dots is considered in this Section. After the camera pose estimation ORB based object localization has been shown indicating accuracy of pose estimation. Some experiments have been shown in Sec. 2 where LED point extraction and matching have been considered.

**5.1. Experimental setup and LED point reconstruction precision.** The localisation system consists of two network cameras. Images from cameras are obtained with 640x480 resolution with 30 fps frame rate in MPEG-4 compression format. Camera placement is represented in Fig. 6.

Fig. 6. Cameras used for object localisation

Camera extrinsic and intrinsic parameter matrices are found by calibrating cameras using OpenCV library. The calibration object for calibration purposes used is LED board that is represented in Fig. 1a. The LED board pattern was chosen arbitrary, because it is assumed that the pattern doesn't affect accuracy.

To estimate accuracy of camera poses the 3D coordinates by calibrating LEDs were estimated using triangulation and these coordinates are compared to the real coordinates.

Figure 7 shows reconstructed camera poses, represented the camera coordinate axis for each camera and the reconstructed 3D coordinates of calibration LEDs are showed. The accuracy of the reconstructed points is given in terms of Euclidian distance (24):

$$d_i(p, p_r) = \sqrt{(x - x_r)^2 + (y - y_r)^2 + (z - z_r)^2}, \quad (24)$$

where $d_i(p, p_r)$ is distance between coordinates of reconstructed and actual points, $x$, $y$, $z$ are corresponding coordinates of actual points and $x_r$, $y_r$, $z_r$ are coordinates of reconstructed points. The RMS error for reconstructed points is 1.7 mm and maximum error for points is 8.5 mm.

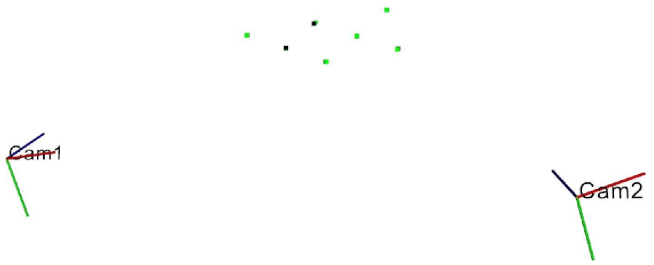


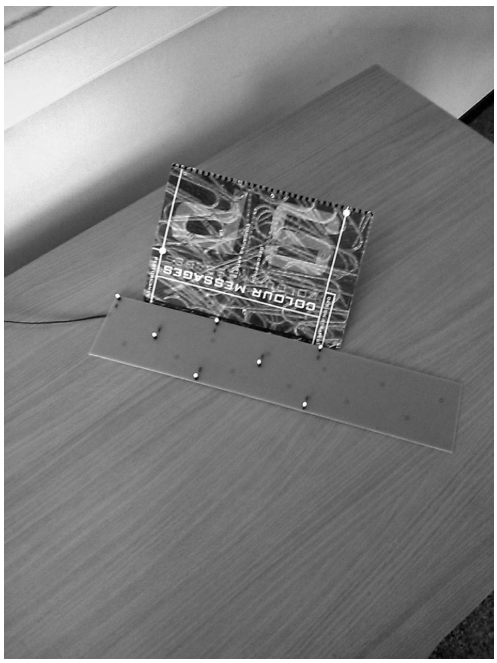

Fig. 7. Camera poses reconstructed via known LED object. Black points are reconstructed coordinates of calibration object, green points are actual coordinates of calibration object

The same LED points have been reconstructed using fundamental matrix (ignoring a priori information about LED point positions) and according to our simulations we could improve results compared to methods with initial calibration only when added error where significantly small (equivalent to some pixels). Cope with LED point coordinate irregularities using fundamental matrix has to be considered for precise enough reconstruction.

In Fig. 8 is shown reconstruction simulation of LED points via fundamental matrix when LED point estimation is perfect.

Object detection for tracking has been implemented using ORB keypoint detection and matching. Keypoints have been extracted from each frame of both cameras and keypoint descriptors have been then matched to previously extracted keypoints of tracking object. If considerable amount of matches have been found in frames of both cameras the object is detected, and it's location can be estimated via triangulation. In Figs. 9 and 10 the detection of object is shown in both cameras. The borders of object have been found by computing perspective transform of borders from the trained image using matched keypoints. After the border has been found the intersection of diagonals for the quadrilateral serve as the tracking point.

a)



b)

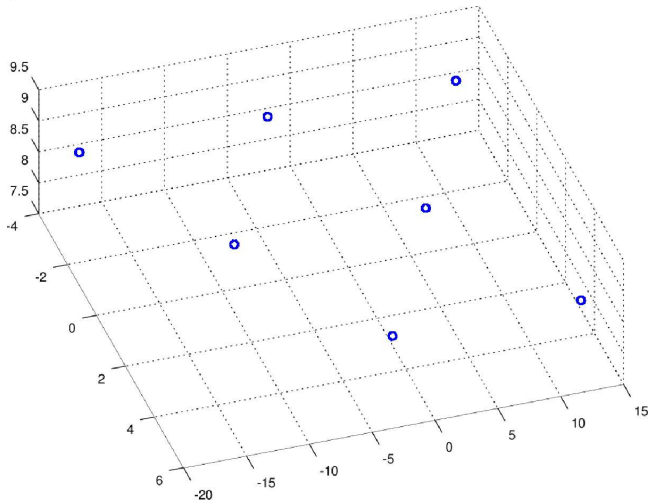



Fig. 8. Computer simulated LED point reconstruction via fundamental matrix

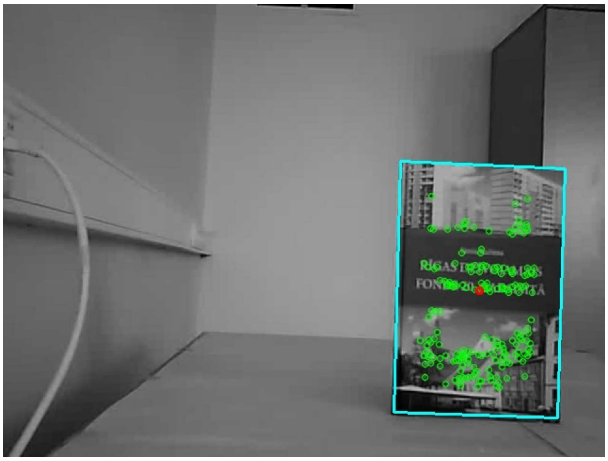


Fig. 9. Left camera image of tracking object

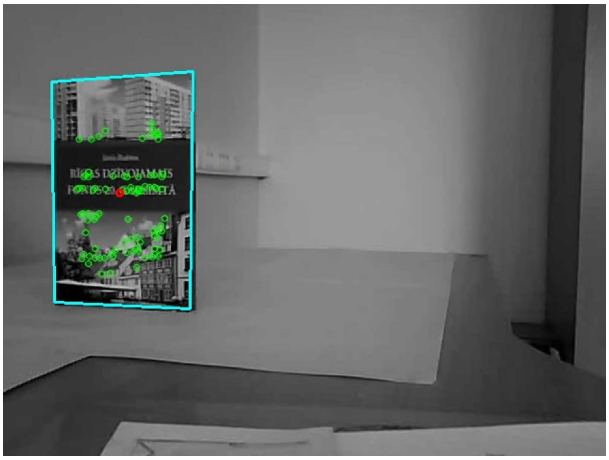


Fig. 10. Right camera image of tracking object

**5.2. Object tracking results.** To evaluate accuracy out of experimental setup for object localization, experiment has been made where object location coordinates estimated with experimental setup has been compared to actual coordinates of objects position. The actual coordinates of object where estimated by placing object on marked coordinate grid. The book shown in Figs. 9 and 10 was used as an object to be localized. The coordinates of the object on the coordinate grid where estimated with accuracy of 0.5 mm. The trajectory of an object was chosen to be in one plane.

Object location has been estimated in 87 locations, and the trajectory for object is shown in Fig. 11. It can be seen that estimated object coordinates are close to actual coordinates of object position. The error metric for accuracy estimation is chosen Euclidian distance between points (24).

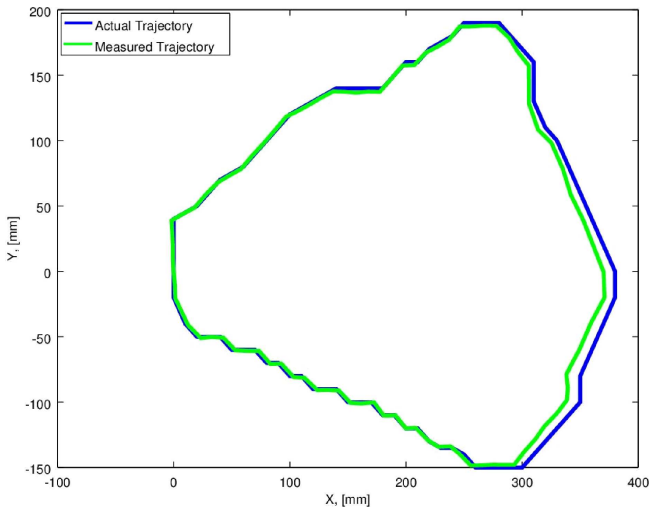



Fig. 11. Comparison of estimated object trajectory with real trajectory

The errors estimated for object positions ranged from 0.7 mm to 12 mm. In Fig. 12 the error for object location with respect to distance from cameras. It can be seen, that there is tendency for error to be bigger at greater distance from camera.

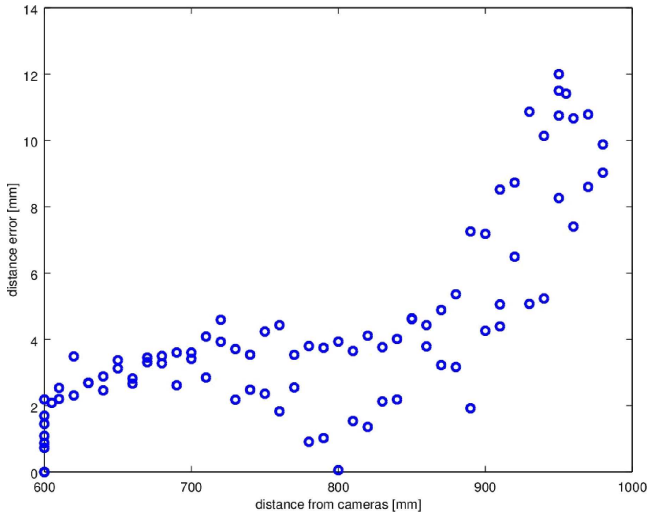



Fig. 12. Object location error with respect to distance from cameras

## 6. Summary and conclusions

The paper considers two basic approaches for camera pose estimation: camera pose estimation via the fundamental matrix and pose estimation via a known object. The static camera case has been considered. Evaluation of camera poses has been done indirectly estimating some object localization and tracking quality. As it were expected the better results provided approach using pose estimation via a known object. Experiments calculating camera poses using ORB points and calculation of the fundamental matrix provided significantly poorer results than usage of defined LED point extraction and pose calculation via a priori known object. After calibration phase the object positions could be estimated with an error ranged from 0.7 mm to 12 mm when the approximate distance between cameras were 50 cm and the object distance from camera base line were 80 cm (see Fig. 12). Developed LED point detection and matching could provide more robust and stable known point extraction. Further development of developed LED point extraction and matching technique has to be continued.

**Acknowledgements.** This work has been supported by funding from EU within the grant No.2013/0008/1DP/1.1.1.2.0/13/APIA/VIAA/016.

REFERENCES

[1] A. Zisserman and R. Hartley, "Multiple view geometry in computer vision", Cambridge University Press, Cambridge, 2003.

[2] E. Rijpkema, K. Muthukrishnan, S. Dulman, and K. Langendoen, "Pose estimation with radio-controlled visual markers", *IEEE 7 th Int. Conf. on Mobile AdHoc and Sensor Systems* 1, 658–665 (2010).

[3] F. Haranz, K. Muthukrishnan, and K. Langendoen, "Camera pose estimation using particle filters", *IEEE Int. Conf. on Indoor Positioning and Indoor Navigation* 1, 1–8 (2011).

[4] M. Faessler, E. Mueggler, K. Schwabe, and D. Scaramuzza, "A Monocular pose estimation system based on infrared LEDs", *IEEE Int. Conf. on Robotics and Automation (ICRA)* 1, 907–913 (2014).

[5] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF", *http://www.vision.cs.chubu.ac.jp/CV-R/pdf/Rublee_iccv2011.pdf*.

[6] E. Rosten, "FAST corner detection", *http://www.edwardrosten.com/work/fast.html*.

[7] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: binary robust independent elementary features", *Computer Vision ECCV 2010, Lecture Notes in Computer Science* 6314, 778–792 (2010).

[8] R.W. Hamming, "Error detecting and error correcting codes", *The Bell System Technical J.* 29 (2), 147–160 (1950).

[9] J.E. Bresenham, "Algorithm for computer control of a digital plotter", *IBM Systems J.* 4 (1), 25–30 (1965).

[10] P.L. Rosin, "Measuring corner properties", *Computer Vision and Image Understanding* 73 (2), 291–307 (1999).

[11] X. Armangue and J. Salvi, "Overall view regarding fundamental matrix estimation", *Image and Vision Computing* 21 (2), 5–220 (2003).

[12] R.I. Hartley, "Estimation of relative camera positions for uncalibrated cameras", *Proc. Eur. Conf. Computer Vision* 1, CD-ROM (1992).

[13] O.D. Faugeras, "What can be seen in three dimensions with an uncalibrated stereo rig?", *Lecture Notes in Computer Science* 588, 563–578 (1992).

[14] G. Strang, "Introduction to Linear Algebra", Wellesley-Cambridge Press, Cambridge, 2009.

[15] H.C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections", *Nature* 293 (5828), 133–135 (1981).

[16] M. Dhome, M. Richetin, and J.-T. Lapreste, "Determination of the attitude of 3D objects from a single perspective view", *IEEE Trans. Pattern Analysis ad Machine Intelligence* 11 (12), 1265–1278 (1989).

[17] R. Horaud, B. Conio, O. Leboulleux, and B. Lacolle, "An analytic solution for the perspective 4-point problem" *Computer Vision, Graphics, and Image Processing* 47 (1), 33–44 (1989).

[18] R.M. Harlick, D. Lee, K. Ottenburg, and M. Nolle, "Anallysis and solutions of the three point perspective pose estimation problem", *Conf. Computer Vision and Pattern Recognition* 1, 592–598 (1991).

[19] L. Quan and Z. Lan, "Linear N-point camra pose determination", *IEEE Transaction on Pattern Analysis and Machine Intelligence* 21 (7), 774–780, (1991).

[20] B. Triggs, "Camera pose and calibration from 4 or 5 known 3D points", *Int. Conf. Computer Vision* 1, 278–284 (1999).

[21] P.D. Fiore, "Efficient linear solution of exterior orientation", *IEEE Trans. on Pattern Analysis and Machine Intelligence* 23 (1), 140–148 (2001).

[22] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: an accurate O(n) solution to the PnP problem", *Int. J. Computer Vision* 81 (2), 155–166 (2009).

[23] D.G. Lowe, "Fitting parameterized three-dimensional models to images", *IEEE Trans. and Pattern Analysis and Machine Inteligence* 13(5), 441–450 (1991).

[24] D. DeMenthon and L.S. Davis, "Model-based object pose in 25 lines of code", *Int. J. Computer Vision* 15, 123–141 (1995).

[25] C.-P. Lu, G.D. Hager, and E. Mjolsness, "Fast and globally convergent pose estimation from video images", *IEEE Trans.* 22 (6), 610–622 (2000).

[26] M.A. Fischler and R.C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Comm. Assoc. Comp. Mach.* 14 (6), 381–395 (1981).

[27] P.D. Sampson, "Fitting conic sections to 'very scattered' data: an iterative refinement of the Bookstein algorithm", *Computer Vision, Graphics, and Image Processing* 18, 97–108 (1982).