Calibrationless sensor fusion using linear optimization for depth matching

László Havasi[1], Attila Kiss[1,2], László Spórás[1,3], and Tamás Szirányi[1,3]

[1] is with the Distributed Events Analysis Research Laboratory, Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary
contact: {*laszlo.havasi,attila.kiss,tamas.sziranyi*}@sztaki.mta.hu
[2] is with the Department of Computer Science of L. Eötvös University, Faculty of Science, Budapest, Hungary
[3] is with the Peter Pazmany Catholic University, Budapest, Hungary [⋆]

Abstract. Recently the observation of surveillanced areas scanned by multi-camera systems is getting more and more popular. The newly developed sensors give new opportunities for exploiting novel features.
Using the information gained from a conventional camera we have data about the colours, the shape of objects and the micro-structures; and we have additional information while using thermal camera in the darkness. A camera with depth sensor can find the motion and the position of an object in space even in the case when conventional cameras are unusable. How can we register the corresponding elements on different pictures? There are numerous approaches to the solution of this problem. One of the most used solutions is that the registration is based on the motion. In this method it is not necessary to look for the main features on the pictures to register the related objects, since the features would be different because of the different properties of the cameras. It is easier and faster if the registration is based on the motion. But other problems will arise in this case: shadows or shiny specular surfaces cause problems at the motion.
This paper is about how can we register the corresponding elements in a multi-camera system, and how can we find a homography between the image planes in real time. So we can register a moving object in the images of different cameras based on the depth information.

1 Introduction

In crowded places it is difficult to track correctly individual people from a single point of view with a conventional camera. That is why we started to investigate multi-camera systems for a better tracking algorithm. We try to fit different camera images onto one common image plane in order to gain a 3-dimensional

[⋆] This work has been supported by the European Communitys Seventh Framework Programme (FP7-SEC-2011-1) under grant agreement n. 285320 (PROACTIVE project). The research was also partially supported by the Hungarian Scientific Research Fund (grants No. OTKA 108947 and OTKA 106374).

picture of the investigated territory in real time. Plenty of research results exist in the literature in connection with this topic when one tries image registration or fitting objects. It's more difficult to distinguish different objects on an image because of the occlusion. If we have more images from different angles about the same screen then we will have more information about that fixed place and the objects in it. By setting an image as the reference image, we can identify the ground plane on every image. Hence we can match the ground points of the objects. If we would shift the ground plane in the direction of its normal vector then we would cut the objects in the same position in all the images. Furthermore we could get more information from a view from above: we could get the spatial position of each object, if we could register the images in different planes together. This plane might be shifted to scan the intersections in the common space of the views [10].
A common plane of all views has an outstanding role: [12] assumes that a reference plane is visible in all views, and shows for this case that lines and cameras, as well as, planes and cameras have a linear relationship. Consequently, all 3D features and all cameras can be reconstructed simultaneously from a single linear system, which handles missing image measurements naturally.
We used the idea of Khan et. Al [10] and developed a method that applies planar homographic occupancy. The present method is developed for multiple planes parallel with the center line of a depth sensor applied together with at least one of the cameras (this is different from the method of Khan and Shah). We try to fit the different image planes onto one screen and try to find different homographies for this, in order to shift these planes into different depth values, then getting an exact match of corresponding moving objects on different image planes. We use a combinatorial algorithm to find these homographies applying a linear programming algorithm from the literature of combinatorial optimization. This algorithm is developed with applying an optimization method based on the well-known golden ratio [9].
The aforementioned method can be used if we have more cameras than two or three, for example on a football match we can follow the movement of the players with this algorithm. In practice there are few events where multiple-camera systems are necessary. In these cases there are different methods for point to point registration and we chose the most efficient one according to our problem. Camera observation is usually integrated into busy places hence there exists motion in all the images. We can make motion mask statistics about these movements, related to a reference image. By detecting motion on the pixel array of this image we make a conditional statistics to the motion masks of the reference image and to all the other images. We will have different motion statistics for each pixel of each reference image belonging to different cameras exactly as many statistics as many cameras we used to integrate into the system. [15] (see Fig. 1) The structure of this paper is first we introduce an algorithm for the image matching. In particular we will talk about the motion detection, the fitting of the image planes and the mathematical problems that we found and the mathematical background of our solution. In the next section we will mention

some experimental results. And we will mention some future directions of this research, some possible ways to carry on with this work.

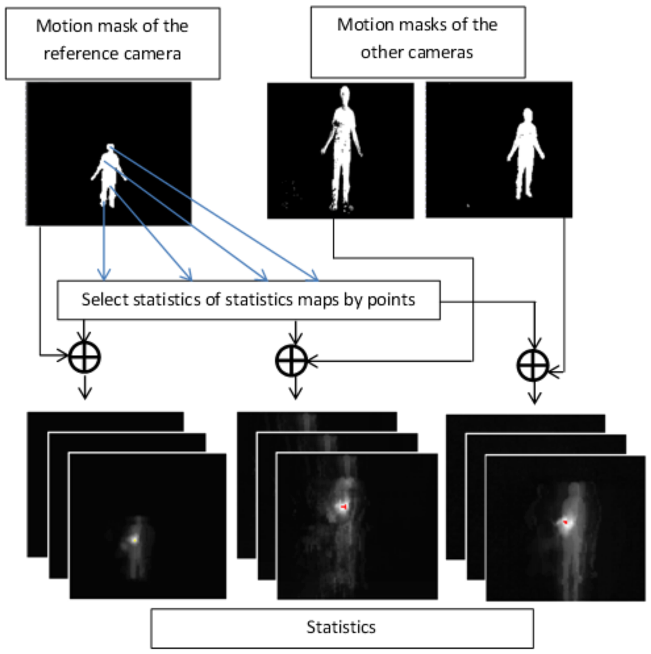

Motion mask of the reference camera
Motion masks of the other cameras
Select statistics of statistics maps by points
Statistics

Fig. 1: Images of the first row are the motion masks. We build the motion statistics maps, assigning places where motion exists (white pixel) on the reference motion mask. Then we add all motion masks to statistics belonging to it, including the reference motion mask. If we have enough number of measurements, we can pair correlated moving pixels related to the reference camera. This conditional statistics of motion map consists of Gaussian distributions. The most probable pair of a point of the probabilistic map is that one of the highest probability. In the probabilistic pairing we assign not only one point but a small neighborhood around it. The red areas of the statistics represent the neighborhood around the maximum.

2 Image Matching

The main steps of the algorithm for image matching based on the depth values are as follows:

1. Own motion detection algorithm for the depth sensor.
2. From the motion masks, we can get the coordinates of different points on all images, and we will show how it becomes possible to register the corresponding points.

3. What about the homography? Different problems are coming up if these corresponding points are belonging to planes that are orthogonal to the ground plane and have different depth values. How can we deal with these problems with a depth sensor?

4. Now we have a homography to project corresponding points from the same plane onto a different plane. We give a method to modify this homography in order to calculate different homographies into parallel planes to the original one.

5. We need to define the variable γ that doesn't change in a linear manner while the change of the depth values is linear.

2.1 Depth motion detection

For the cameras without depth sensors we use the built in class of OpenCV. This class is BackgroundSubtractorMOG. Unfortunately this motion detector does not give perfect results, furthermore the output will be too noisy. If we have more moving objects in occlusion with each other, then we are not able to find their original depth values. That was the motivation why we developed an algorithm for depth sensors, that determines the motion from the change of depth values [13]. So we get more reliable and less noisy motion masks from the depth sensor and these are important for further measurements.
The fundamental idea is that the sensor collects data in the first step, and we store them. If we have collected enough data, the algorithm can create an initial background mask from this database. Each pixel of this initial mask will get the average value of the measurements in that pixel. If there is a moving object, then the depth values of this object will be smaller, then the values of the background mask, and the depth values of the moving object will be added to the foreground mask. The algorithm can learn, for example if an object in the foreground mask doesn't move, and the depth values of it are not changing, and these values are lower than a threshold, then the background mask will be updated with these values. Furthermore if the depth values of an object suddenly get larger (after an object has moved away the depth values of the background became larger), then the algorithm updates the average of the depth values to the background mask. [2, 6, 3] (see Fig. 2)

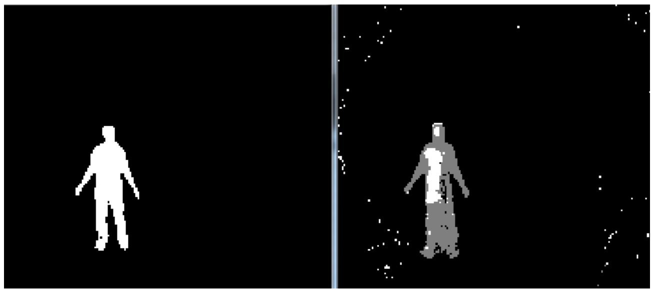


Fig. 2: The image on the right was made with the in-built motion detector class of the OpenCV. We can see, that the motion mask is not perfect, and the image contains noisy pixels.
The image on the left has a good motion mask without noise as the result of the proposed method.

2.2 Corresponding motion statistics

It is necessary to find accurate point pairs of high correspondence in order to obtain a good homography between two image planes. The size and the shape of a moving object may be significantly different in images of different cameras, and matching corresponding points is more difficult and becomes less accurate if we work with more cameras. That was the reason why we create co-motion statistics for each camera. We applied the idea of investigating corresponding motions from a previous work of the authors László Havasi, Tamás Szirányi and our former colleague and co-atuhor Zoltán Szlávik [15]. We need a reference camera, fixing its image plane as the reference plane where we will fit the image planes of the other cameras with the proper homographies. This camera must need a depth sensor, so we used a professional camera with depth sensor, namely Mesa Imaging Swiss Ranger 4000.
If our algorithm detects motion in any pixel of the reference image, then we update the statistics of this pixel position for each camera with the current motion mask [15].
For memory and run-time reasons we re-size the reference image and motion maps to size 80 * 80. For example we set a motion statistics map with 80 * 80 size to each camera, and each element of that map is a statistics with size 80 * 80. Let us suppose that we sense motion in the coordinates (45, 35) of the reference image, then we add the motion mask of the related image to the statistical map of the pixel in the coordinates (45,35) of the reference camera. Trivially we have to re-size the motion masks of the other cameras to size 80*80 too. Experimental results showed that the resizing for this size keeps the sufficient information for the given task.

2.3 Problem with the homography

The cameras are sensing the world from different point of views, so the corresponding objects will have different sizes on their images. We used homography to fit the different images onto a reference image. We need four point pairs to find a homography between two image planes. A point pair is an element from the Cartesian product of the set of the vertices of the reference image plane and another image plane. We are looking for corresponding points on these image planes, but after the fitting the transformation may deform the picture (because the image planes are not have to be coplanar), moreover, the transformation may rotate the whole image plane.
But if the corresponding points would lie in the same plane it would not be sufficient neither. Because the fitting would be accurate only in their plane, but it would be inaccurate in all the other parallel planes. (see Fig. 3) [4, 7, 14]

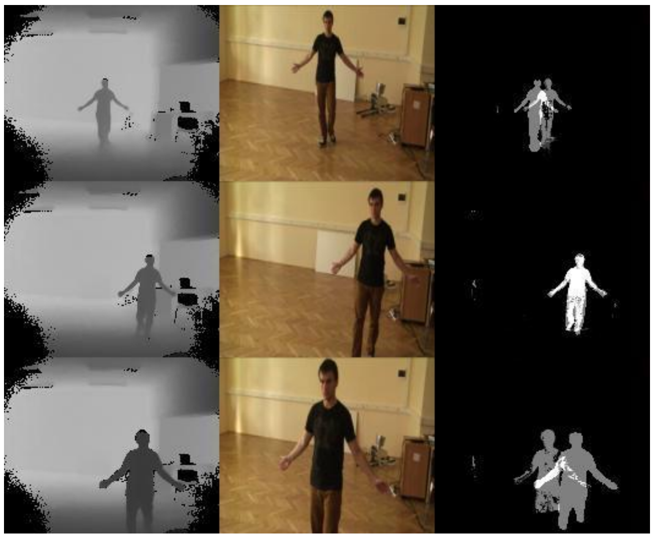
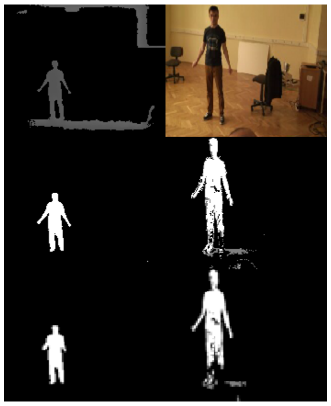


Fig. 3: In these pictures one can see in the first column the outputs of a depth sensor, in the second column the outputs of a conventional camera and in the third column the image after merging the motion masks of these two sensors
First row: The object is placed behind the proper plane of the homography
Second row: The object is placed in the proper plane of the homography
Third row: The object is placed before the proper plane of the homography

Our task is to project these point pairs onto the same image plane and make a procedure that can shift this plane into different depth values. When this procedure is fast enough that will make us able to match the corresponding moving objects on different camera images in real time.
In order to ensure that the corresponding points lie in the same plane, we use

a depth filter on the reference camera to choose that plane on which we want to find a homography. Now the reference camera will perceive the motions only in the environment of the selected depth, hence it will generate motion masks only in that depth. This will result motion statistics that we needed to have for our algorithm. The point pairs that can be achieved from these statistics will be located on the same plane. (see Fig. 4) For further details the reader is referred to: [14]


(a) Object in the plane of the homography

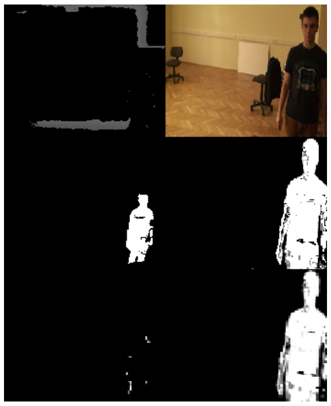

(b) Object in a different plane

Fig. 4: The depth filter: actual distance is: 3,5m from the camera with 0,3m depth range. Valid geometrical depth is 3,2 – 3,8m.
In the figures in the first column there is an image from the depth sensor when we apply a depth filter on it and the motion mask of the depth sensor without the depth filter and the intersection of the two images in the first two rows, from top to bottom respectively. In the second column there is an image from a conventional camera and the motion mask of that image and a eroded shrunk image from the motion mask of the conventional camera, from top to bottom respectively. The bottommost pictures are applied during the construction of the statistics maps of the co-motion.

2.4 Pushing the plane of the homography depending on depth

After we have found a good homography for the points in the same depth, we only have to solve the parallel shifting of homography calculation for different source planes.
We used the fusion of multiple planes to increase robustness and accuracy of our method. Our method performs fusion of different image planes onto one reference plane. One can see that conventional feature correspondence-based methods are not feasible for homography calculus among image planes. For example if we have a homography $H_{p,q}$ induced by a reference plane r between to different

views p and q then the homography $H_{p,q}$ induced by a plane s parallel to r is given by the following formula: (for further details the reader is referred to [10]:

$$H_{psq} = (H_{prq} + [0|\gamma v_{ref}])(I_{3x3} - \frac{1}{1+\gamma}[0|\gamma v_{ref}]) \tag{1}$$

Where v_{ref} is the vanishing point of the normal direction and γ is a scalar multiple that has to control the distance between the parallel planes. Typically we use the plane orthogonal to the center line of the depth sensor of the reference camera as the reference plane and the parallel direction with that line as the reference direction. Here the homographies were determined with SIFT [11] feature matches and using the RANSAC algorithm [8].

2.5 Defining the γ parameter

If the value γ approaches the $\pm\infty$, then the homography H_{psq} converges to stable states. Experimental results showed that the final value of H_{psq} is well approximated on the interval [-20, 20]. (see Fig. 5)

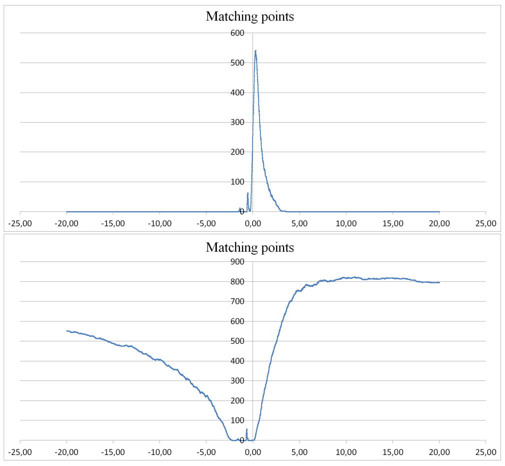


Fig. 5: We used two different motion masks. One is the motion mask of the reference camera the other one is the motion mask of another camera that we want to fit. The first graph is about the case when the matching function is calculated in an image plane with larger depth value than the the reference plane has. The other graph is about a matching function in the reference plane. The axis X shows the value of γ, the axis Y is the amount of matching points after the fitting. A coordinate on the motion mask of the reference image will be matched if our algorithm senses motion in that coordinate and in that pixel there is motion in the motion mask of the other camera too. Our algorithm is looking for a fitting that maximizes the amount of the matched points between two images.

After we detected a homography we wanted to find a method to refresh the optimal value of the scalar multiple γ for the best fitting if we shift our reference plane into different depth. We applied a combinatorial method to find this optimal γ. First we calculated a range small enough to start the search in with random evaluating an error function indicating the value of correct fitting. After that we applied a convex optimization method, a specially modified version of the well-known interval halfing method, applying the golden ratio values for the shrinking of the interval. We could do this because we managed to prove that this error function using this value γ is convex with probability 1. We made some tests and it resulted that our algorithm runs in real time. In comparison with other methods, there are some algorithms trying to solve problems similar to our, for example in the article [1] we find similar arrangement of equipments, but that solution takes about a minute per frame to get depth information, which is much slower than our present real-time approach.

3 Experimental results

We used this system for an indoor scene because our camera with the depth sensor is unsuitable for outdoor recording, the algorithm itself is applicable in outdoor territories too. However the base distance of cameras is a main parameter of our algorithm. If this distance is getting larger then the imprecision of the matching will be bigger. Furthermore the response time will be slower, if more cameras are integrated into the system (see Fig. 6 and Fig. 7), or we increase the resolution of the cameras.
Currently the system works in case of only one object (see Fig. 9 and Fig. 10). However if we can collect enough information with this single object about the γ values of different depth, then the matching algorithm can be omitted, simply we can search in a database to find the value γ of the current depth. This method eliminates the need for a matching algorithm handling more objects. Since we have a database of pairs of depth and γ values, it is possible to estimate from the picture of depth camera that one or more objects moving in front of the cameras. After the database has been built up, we are left only a few possibilities to choose the proper depth value and the object helps us to determine the exact depth value because the matching of the objects would not be possible in another plane with the given value γ.

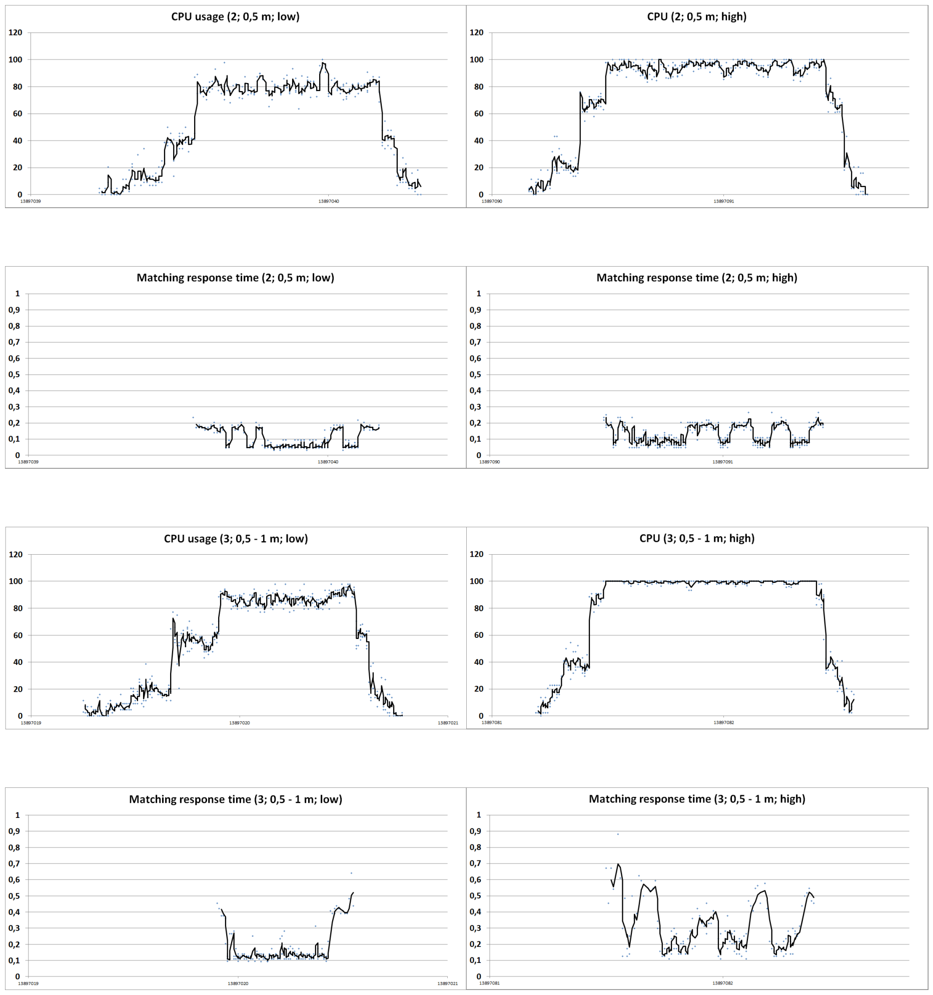


Fig. 6: Figures in the first and third lines show the CPU usage where the axis x is the time (CPU time) and the axis y is the CPU utilization in percentage. We started the measurements from the idle state of the CPU. The steps of each measurement were the following: start the sampling of the cameras one by one, start the generating of the motion masks, start the fitting phase (two different types of result, successful or unsuccessful), stop the fitting phase, stop the generation of the motion masks, stop the sampling of the cameras. Figures in the second and the fourth lines show the response time of the algorithm where the axis x is the time (CPU time) and the axis y is the response time in second. The first set of figures of the first two rows are belong together and their axis x is uniformly scaled in order to get a better view. The different parametrization of measurements are indicated in the title of the figure in the form (n, d, r) where n stands for the number of cameras we used, d stands for the distance of the cameras from the reference camera and r stands for the resolution we applied (high or low). The resolution of the Mesa camera is (176x144), the two different resolutions we used for the Axis Network camera are (320x180) and (800x450) and for the Axis Thermal Network camera are (240x180) and (480x360).

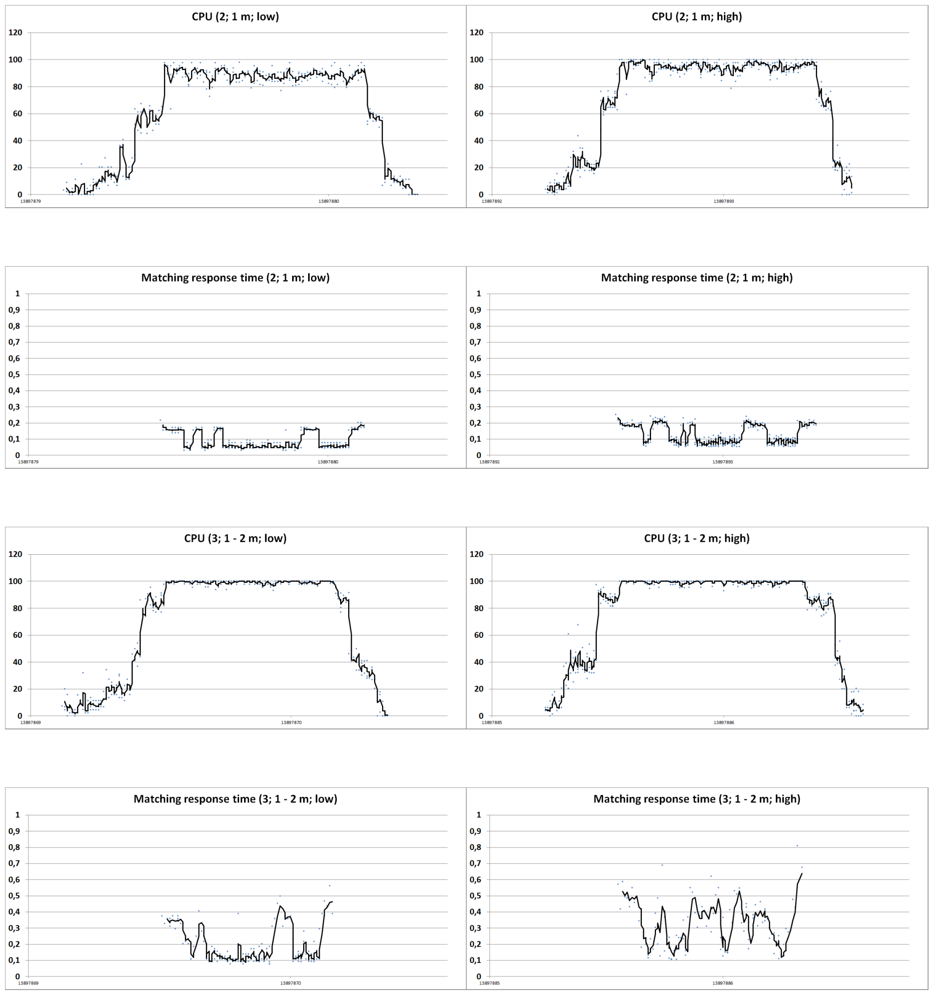


Fig. 7: These figures show what happens if we increase the distances between the cameras and the reference camera.

We made a figure to compare the response times of the different cases. See Fig. 8.

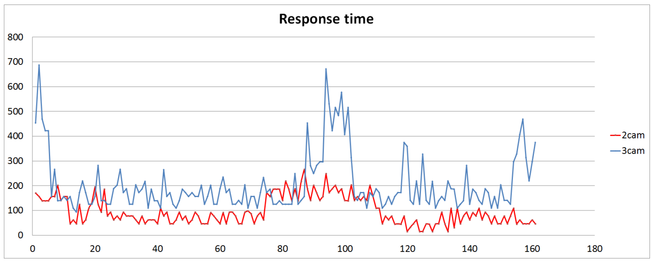


Fig. 8: Response time of the matching with 2 or 3 cameras
Red line is the measurement of 2 cameras, blue line is the measurement of 3 cameras. Axis X is the serial number of frames, axis Y is the running time in millisec. If the algorithm finds a good matching, the response time will be under 100 millisec in case of 2 camera (between 30 - 70), if do not find, then this time will be larger (between 80 - 100). In the other case these time values are higher.

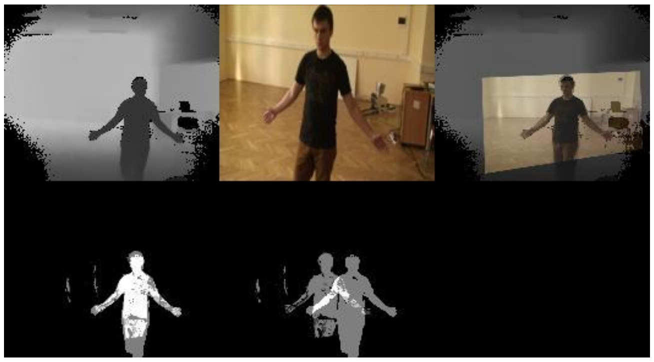


Fig. 9: The result of 2 cameras
We used a depth sensor (Mesa Imaging Swiss Ranger 4000) and a visible spectrum camera (Axis Optical Camera)
The first two images are the raw data from cameras in the first row. The first image of the second row is the matching with a plane gained by pushing the plane of the homography, second image is the original plane of the homography without pushing. Third image of the first row is the result on the raw data with the matching homography.

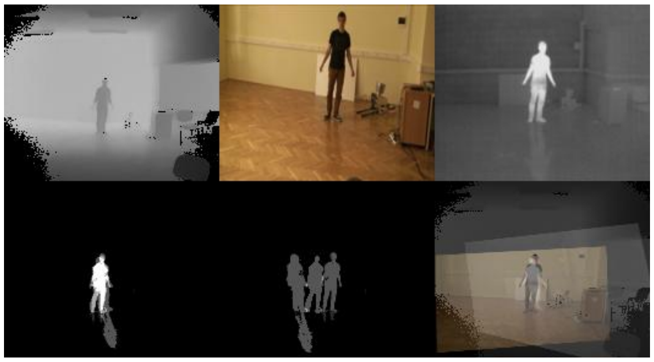


Fig. 10: The result of 3 cameras
We used a depth sensor (Mesa Imaging Swiss Ranger 4000), a visible spectrum camera (Axis Optical Camera), and a thermal camera (Axis Thermal Camera) The first three images are the raw data from cameras. The first image of the second row is the matching with a plane gained by pushing the plane of the homographies, second image is the original plane of the homographies without pushing. Third image of the second row is the result on the raw data after the matching with the homographies.

4 Conclusions

The paper presents a new method for camera fusion. We used co-motion statistics to register the image planes together, and a combinatorial algorithm to find depth values to each image planes. Our idea was to combine these methods and to use the golden ratio in the convex optimization step. It may provide a solution to application cases like safety systems operating with more cameras. It also helps to make a conclusion in case of the breakdown of a camera, caused that by an intruder target person, if the rest of the cameras perceive him. If the system has already finished the learning phase, then the remaining cameras are enough to recognize the target person on their images and determine the position of the intruder target person in the image of the camera that was broke down.

References

1. Jeroen van Baar, Paul Beardsley, Marc Pollefeys and Markus Gross, 2012: Sensor Fusion for Depth Estimation, including TOF and Thermal Sensors, *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), Second International Conference on. IEEE*, 472–478.
2. Olivier Barnich and Marc Van Droogenbroeck, 2011: ViBe: A universal background subtraction algorithm for video sequences, *In IEEE Transactions on Image Processing*, June 2011, 20, 6, 1709–1724.

3. Olivier Barnich and Marc Van Droogenbroeck, 2009: ViBe: a powerful random technique to estimate the background in video sequences, *In International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2009)*, April 2009, 945–948.
4. Harold Scott Macdonald Coxeter, *Projective geometry*, Springer, 2003.
5. Antonio Criminisi, Ian Reid and Andrew Zisserman, 2000: Single View Metrology, *International Journal of Computer Vision*, November 2000, 40, 2, 123–148.
6. M. Van Droogenbroeck and O. Paquot, 2012: Background Subtraction: Experiments and Improvements for ViBe, *In Change Detection Workshop (CDW), Providence, Rhode Island*, June 2012, 1709–1724.
7. Richard I. Hartley, 1999: Theory and Practice of Projective Rectification, *International Journal of Computer Vision 35 2*, 115–127.
8. Richard Hartley and Andrew Zisserman, 2002: *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, United Kingdom
9. Tibor Illés, 2011: *Nonlinear Optimization*, University Lecture Notes, L. Eötvös University of Science, Budapest, Hungary, (in Hungarian)
10. Saad M. Khan and Mubarak Shah, 2009: Tracking Multiple Occluding People by Localizing on Multiple Scene Planes, *IEEE Transactions On Pattern Analysis And Machine Intelligence*, March 2009, 31, 3, 505–519,
11. David Lowe, 2004: Distinctive Image Features from Scale Invariant Keypoints, *International Journal of Computer Vision*, 91–110.
12. Carsten Rother, 2003: Linear Multiview Reconstruction of Points, Lines, Planes and Cameras Using a Reference Plane, *Proceedings. Ninth IEEE International Conference on Computer Vision*, 2, 1210–1217.
13. Jamie Shotton, Ross Girshick, Andrew Fitzgibbon, Toby Sharp, Mat Cook, Mark Finocchio, Richard Moore, Pushmeet Kohli, Antonio Criminisi, Alex Kipman, and Andrew Blake, 2013: Efficient Human Pose Estimation from Single Depth Images, *In Decision Forests for Computer Vision and Medical Image Analysis*, 175–192.
14. László Spórás, 2013: Sensor fusion of depth sensors, color and thermal cameras with small base distance (Bachelor's Thesis), Peter Pazmany Chatolic University, Budapest, Hungary, (in Hungarian)
15. Zoltán Szlávik, Tamás Szirányi, László Havasi, 2007: Stochastic view registration of overlapping cameras based on arbitrary motion, *IEEE Tr. Image Processing*, Vol.16, No.3, 710–720.