# Five Shades of Grey for Fast and Reliable Camera Pose Estimation

Adam Herout, István Szentandrási, Michal Zachariáš, Markéta Dubská, Rudolf Kajan
Graph@FIT, Brno University of Technology
Brno, Czech Republic
herout@fit.vutbr.cz
http://www.fit.vutbr.cz/research/groups/graph/MF/

## Abstract

*We introduce here an improved design of the Uniform Marker Fields and an algorithm for their fast and reliable detection. Our concept of the marker field is designed so that it can be detected and recognized for camera pose estimation: in various lighting conditions, under a severe perspective, while heavily occluded, and under a strong motion blur.*

*Our marker field detection harnesses the fact that the edges within the marker field meet at two vanishing points and that the projected planar grid of squares can be defined by a detectable mathematical formalism. The modules of the grid are greyscale and the locations within the marker field are defined by the edges between the modules.*

*The assumption that the marker field is planar allows for a very cheap and reliable camera pose estimation in the captured scene. The detection rates and accuracy are slightly better compared to state-of-the-art marker-based solutions. At the same time, and more importantly, our detector of the marker field is several times faster and the reliable real-time detection can be thus achieved on mobile and low-power devices. We show three targeted applications where the planarity is assured and where the presented marker field design and detection algorithm provide a reliable and extremely fast solution.*

## 1. Introduction

For augmented reality applications and other similar problems in computer vision, camera localization within a captured scene is crucial. Camera localization can be done either by using fiduciary markers [6] or without them (by using PTAM [10], keypoint template tracking [16], homography-based [11], etc.). We are dealing with applications and scenes where fiduciary markers are acceptable (see Sec. 5 for examples). At the same time, we require the detection and localization algorithm to be extremely fast (to work in real time on mid-level ultramobile devices) and to localize the camera from a single frame (i.e. without temporal tracking and mapping).

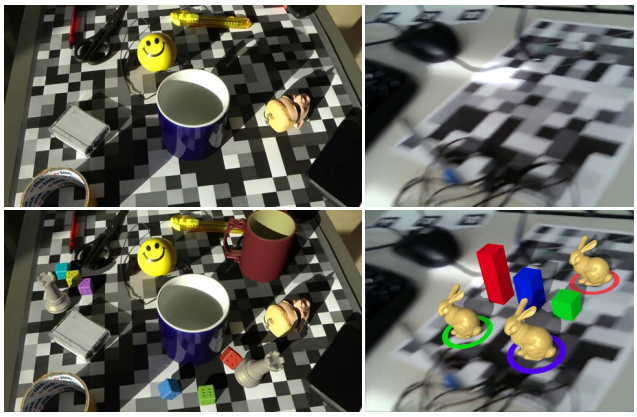

Figure 1. The use of our Marker Field. **left:** MF with occlusion, sharp shadows. **right:** MF with clutter, occlusion, varying lighting, strong blur. **top:** Original image – input to the recognizer. **bottom:** Recognized camera location and augmented scene.

The targeted applications (Sec. 5) allow for perfectly planar markers – placed on a tabletop, a wall, computer screen, etc. The challenge is that the marker must cover a large planar area and, at the same time, it must be reliably detected even from a small visible portion of the marker. Also, the detection must be invariant to high degrees of perspective distortion and to varying lighting conditions (direct light, shadows, different lighting intensities). What marker design and corresponding detection algorithm can meet these requirements and, at the same time, be aesthetically appealing? A step towards the solution of this problem was recently sketched out by Szentandrási et al. [14]. Their Uniform Marker Fields are planar checkerboard fields of largely overlapping markers, shaped as a 4-orientable $n^2$-window array [3]. This overlapping property allows the marker fields to outperform arrays of conventional disjoint markers such as the ARtag [5], ALVAR [1], or CALTag [2]. Marker-based solutions such as ARtag and ALVAR (a number of other similar solutions exists) are using square black-and-white markers with their identity digitally encoded. One

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.182

part of the marker's design is used for the marker's localization (typically the outer black/white rim) and another part is used for distinguishing between individual markers (typically inner content of the square). An array of such individual markers is used to cover a larger planar area. CALtag [2] alters black-and-white with white-and-black (inverse) markers and attaches the markers one to another.

Another approach to overlapping individual detectable windows within a large marker are the Random Dot Markers [17] by Uchiyama et al. They are detecting and tracking fields of randomly displaced dots on a solid background using geometric features. The field of random dots can be also used as a deformable marker [15].

Our solution is based on the Uniform Marker Fields by Szentandrási et al. [14]. In their short work, they used binary De Bruijn tori [3] in a checkerboard as the marker field. In this paper, we propose to use a greyscale grid of squares (instead of a binary one): it offers more edges in the marker field to be detected and, at the same time, a smaller window of the De Bruijn torus is necessary for identifying a unique location. We present here an algorithm for the detection of the greyscale grid of squares. Our algorithm detects the planar projected grid as a single compound object, instead of detecting straight lines and then forming a grid from them. A unique location in the marker field is identified by the edges between the marker field modules. These edges need to be reliably classified – Wald's Sequential Probability Ratio Test [18] is used in order to sample a minimal number of pixels for discerning the edge.

Overall, the detection algorithm has a small data footprint – in the sense that a small fraction of the input image pixels is visited ($\sim 5\,\%$ in our measurements). This allows for the detection to be really fast (1080p frame in 8.8 ms on a Intel Core i5-661 @3.33GHz) and we are presently working on a real-time implementation for ultramobile devices. With this performance (more than $3\times$ faster than ALVAR), our algorithm is still equal or better compared to available solutions in terms of reliability and accuracy (Sec. 4).

We highlight three target applications of this marker field design and detection algorithm (Sec. 5). All these applications (and others as well) can readily use our marker in the scene and they can ensure that the marker is planar. Our measurements show (Sec. 4) that our marker field design and detection algorithm outperforms the existing solutions for this class of camera pose estimation problems.

## 2. Shades of Grey – The Marker Field Design

*Aperiodic 4-orientable binary $n^2$-window arrays* [14, 3] are matrices $A = (a_{ij} \in \{0, 1\})$, where each square subwindow $A_{rc}$ of dimensions $n \times n$ appears only once, including all four rotations. If any of the windows appeared more than once, we would be speaking of a *conflict* – either a mutual conflict between two different windows (possibly rotated) or a self-conflict, where a window is self-similar after rotation. Szentandrási et al. [14] interpret such an array as a black-and-white checkerboard and propose to use it as a marker field for augmented reality. The unique $n^2$-windows largely overlap. Thanks to this overlap, only a small fraction of the marker field must be visible in order to be detected and recognized.

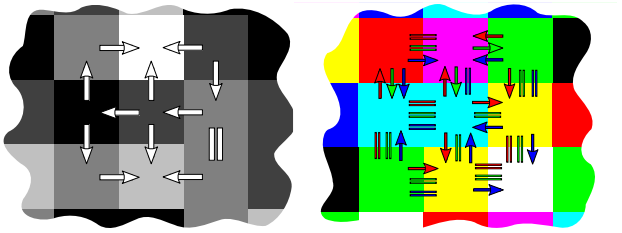

Figure 2. A fragment of the marker field. **left:** Five shades of grey. **right:** 8 different colors. Arrows across the edges illustrate the observable gradient which describes an individual line. In color, multiple gradients can be observed at one edge (e.g. RGB).

We work with grayscale or color $k$-ary marker fields ($a_{ij} \in \{0, \ldots, k-1\}$, Fig. 2). However, in comparison with binary marker fields the absolute greyscale or color values of the grid modules cannot be reliably discerned under varying lighting and camera conditions. That is why we use the *edge gradients between the modules* for localization within the marker field. Horizontal (1) and vertical (2) edge gradients are defined as:

$$e^{\rightarrow}_{ij} = a_{i,j+1} - a_{ij}, \tag{1}$$

$$e^{\downarrow}_{ij} = a_{i+1,j} - a_{ij}. \tag{2}$$

The absolute value of the edge gradient is also hard to recognize reliably and thus only the basic character of the edge is used for recognition: $\operatorname{sgn} e^{*}_{ij} \in \{-1, 0, +1\}$. The $n^2$-window used for localization within the marker field then is (Fig. 2):

$$E_{rc} = (e^{\rightarrow}_{rc}, \ldots, e^{\rightarrow}_{(r+n-1,c+n-2)}, e^{\downarrow}_{rc}, \ldots, e^{\downarrow}_{(r+n-2,c+n-1)}) \tag{3}$$

Synthesis of the marker field is done in a manner similar to the genetic algorithm sketched out by Szentandrási et al. In our case, the fitness function must also reflect the quality of edges between the modules – edges with higher absolute value $|e_{ij}|$ are preferred.

## 3. Small Footprint Detection & Recognition of Planar Greyscale Grids of Squares

This section describes the algorithm for detection of the greyscale checkerboard-like marker field. This algorithm supposes that the grid of squares is planar and projected by a perspective projection. The experiments (Sec. 4) show that this condition is fulfilled enough in realistic scenes observed by standard cameras. Thanks to this assumption, the

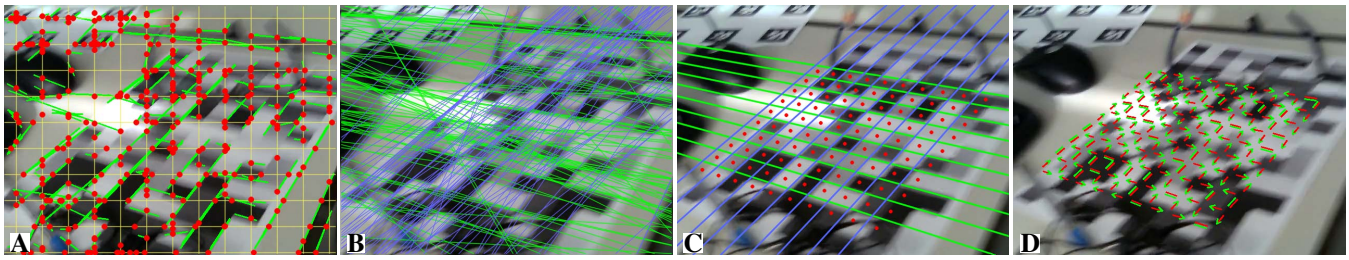


Figure 3. Detection of the greyscale grid of squares. **A:** The image is processed in sparse scanlines. On each scanline, edges are detected (Red) and extended to edgels (Green) by iteratively finding further edge pixels in the direction perpendicular to the gradient. **B:** The edgels are grouped into two dominant groups using RANSAC; two vanishing points are computed by hyperplane fitting. **C:** Based on the vanishing points, the optimal grid is fitted to the set of the edgels (orange dots denote the estimated centers of grid modules). **D:** Edges between the modules are classified (Sec. 3.2). Note that despite the blur, lighting and occlusion in the image, the camera is localized correctly (Fig. 1).

algorithm is very efficient: the fraction of visited pixels (the algorithm's "pixel footprint") within an average input image is very small (Sec. 4.2).

### 3.1. Greyscale Grid Detection

Conventional marker detectors typically rely on first detecting the bounding borders [9, 5] of the markers by finding the contours in a thresholded image and choosing shapes consisting of four straight-line contours. Our uniform marker field does not distinguish between marker design features intended for general *marker detection* and features for marker *identification*. Grid modules serve simultaneously as the detection and identification features. The motivation for this approach is to better use the marker field's surface: the localization features are much denser in the field, while still preserving the identification capabilities.

The algorithm performs the following three main steps (Fig. 3):

**1. Extraction of edgels** (edge element or edge pixel; term borrowed from Martin Hirzer [8]) – typically, the algorithm extracts around one hundred straight edge fragments in the whole image. The image is processed in sparse horizontal and vertical scanlines (Fig. 3A). When a video input is being processed, the detected edgels are filtered based on the previous detected position of the marker field. In the tests we used a simple rectangular mask to filter out the edges outside the area corresponding to the previously detected marker field.

**2. Determining two dominant vanishing points** among the edgels (Fig. 3B). Using homogeneous coordinates for the vanishing point $\mathbf{v}$ and the pencil of lines $\mathbf{l}_i$, all the lines are supposed to be coincident with the vanishing point, i.e.

$$\forall i : \mathbf{v} \cdot \mathbf{l}_i = 0. \tag{4}$$

The coordinates of the lines in the real projective plane form a 3D vector space without an origin (with an equivalence relation). Points of the real projective plane correspond to hyperplanes passing through the origin, so the vanishing point can be found by fitting a hyperplane through all the lines (extended edgels) observed in the pencil. The line vectors $\mathbf{l}_i$ are scaled so that each one's magnitude corresponds to the edgel length. In this way, the longer and more reliable edgels are favored. The hyperplane's normal is found as the direction of the least variance by eigendecomposition of the correlation matrix

$$C = (\mathbf{l}_0 \dots \mathbf{l}_N)(\mathbf{l}_0 \dots \mathbf{l}_N)^T. \tag{5}$$

Since matrix $C$ is $3 \times 3$ and symmetric, decomposition can be computed very efficiently.

**3. Finding the grid of marker field edges** as two groups (*pencils*) of regularly repeated lines coincident with each vanishing point. Two vanishing points $\mathbf{v}_1, \mathbf{v}_2$ define the horizon ($\mathbf{h} = \mathbf{v}_1 \times \mathbf{v}_2$). Marker edges of one direction can be computed using the horizon as ($\hat{\mathbf{x}}$ denotes normalized vector)

$$\mathbf{l}_i = \hat{\mathbf{l}}_{base} + (ki + q)\,\hat{\mathbf{h}}, \tag{6}$$

where $\mathbf{l}_{base}$ is an arbitrarily chosen base line through the vanishing point, different from the horizon [13]. Parameter $k$ controls the line density and $q$ determines the position of the first line. A good simple choice for $\mathbf{l}_{base}$ is a line through the center of the image (and through the vanishing point).

In order to find $k$ and $q$, the value of $(ki+q)$ is calculated for every line (extended edgel) of the input group. These values are clustered by simplified mean-shift and median difference between cluster candidates. (The mean-shift box kernel size with normalized image coordinates in our tests was $w = 0.05$.) Each cluster is assigned an $i$ and then overall optimal $k$ and $q$ are found by linear regression (Fig. 3C, blue and green lines).

For simplicity, the algorithm description supposes that a significant portion of the input image is covered by the marker field. However, steps 2 and 3 of the algorithm are conditionally applied on rectangular parts of the image (quarters, ninths); in high-resolution images, the marker field is thus found even if it covers an arbitrary fraction of the camera input.

### 3.2. Edge Classification

When only a small fraction of the marker field is visible, it is crucial that the edge gradients (Eq. 1 and 2) are recognized correctly. Their recognition can be challenging due to motion blur, uneven lighting conditions, etc. (Fig. 4).

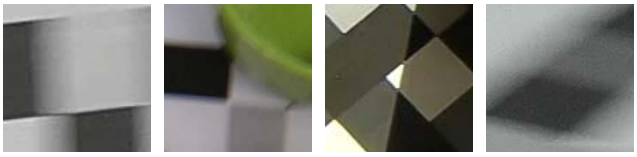

Figure 4. Examples of problematic edges within the pictures of the marker field. The edges are classified by deterministically sampling a varying number of pixels within the neighboring modules. Wald's SPRT is used to discern the edge by using a minimal number of such samples.

In order to correctly classify an edge, given the locations of the neighboring marker field modules, our algorithm samples pixels from the edge's vicinity. If a small number of samples suffices to decide an edge either way ($-1, +1$; see Sec. 2), the decision is made, otherwise more pixels are sampled. If an edge cannot be confirmed, the location between the modules is treated as a place without an edge: $e^*_{ij} = 0$. The stopping criterion is given by Wald's sequential probability ratio test [18], which is proven to be the optimal sequential test for this purpose.

### 3.3. Localization Within the Marker Field

The sub-window described by edges $E_{rc}$ is formulated as a vector of scalars in (3). This vector can be used as a key to a **hash table**. Values in the table represent locations in the marker field (two discrete coordinates in the terms of grid modules; enumerated orientation $0°\,/\,90°\,/\,180°\,/\,270°$). An absent record in the hash table means a wrongly recognized fragment of the marker field. Hash tables are implemented fairly efficiently in today's programming languages.

Instead of using a readymade hash table, we prefer to create a **decision tree**. When a compact piece of the marker field is detected in an input image, the edges are classified and used for traversing the tree. A central edge in the detected cluster of edges decides the root node, and surrounding edges follow in a predefined order (Fig. 5). Any cluster of neighboring edges is recognized by the tree – the leaf node would either define the cluster's location and orientation within the marker field or reject the cluster of edges as invalid (due to misdetection). Constructing a deeper tree implies that a larger cluster of edges is used for localization within the field. This allows for larger marker field resolutions. By using a larger number of deciding edges, the tree can also be constructed **fault-tolerant** – the tree nodes can tolerate one or more falsely classified edges.

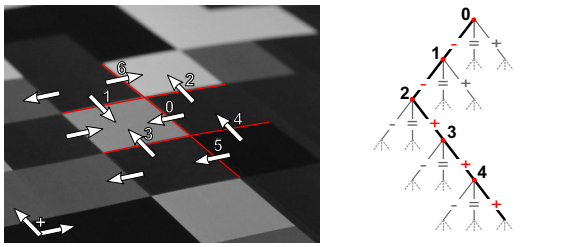


Figure 5. The decision tree used for localization in the marker field. **left:** A compact cluster of edges detected in the image. Edges are numbered in a predefined order relative to a selected "root" edge (0). **right:** Decision tree – the leaves are either *invalid* or contain a location + orientation to the marker field.

### 3.4. Corner Search and Iterative Refinement

For a precise camera pose estimation we find all possible corners in the marker field (with a sub-pixel precision). The corners of the grid of squares are projected from the detected overall position and iteratively searched for in the neighborhood. Based on the marker field layout, the algorithm knows each corner's appearance including its rotation and searches for such a particular pattern. This helps mostly in cases when the image is motion blurred, the marker is not perfectly planar, or noise in the edgel data cause the grid not to fit the edges precisely. Another way of improving the precision of the pose estimation accuracy is to iteratively search for correct corners in the marker field in the image space using back-projection. In the tests we use both of the aforementioned improvements.

## 4. Experimental Results

We compare our solution to ALVAR [1] as the most mature available ARToolKit follower (ARtag is no longer publicly available) supporting arrays of disjointed square markers. The other baseline is the Random Dot Markers (RDM) [17] as an alternative "marker field" solution, where individual localization markers overlap in the field. We performed identical experiments with CALtag [2] as well, but its results were always worse than ALVAR and it is much slower (written in Matlab), so we omit CALtag's results from this paper.

For comparing our solution with the alternatives, we shot videos of side-by-side markers (Fig. 6). The marker fields have comparable (as much the same as possible) dimensions and resolution of the individual markers ($n^2$-windows vs. ALVAR individual markers vs. RDM's sub-markers) and the movement is simple and well-defined to ensure fairness in the comparison (see supplementary material for examples of the videos).

### 4.1. Success Rate and Precision

Fig. 7 shows the estimated camera pose in graphs for different videos for ALVAR and UMF. In order to evaluate the

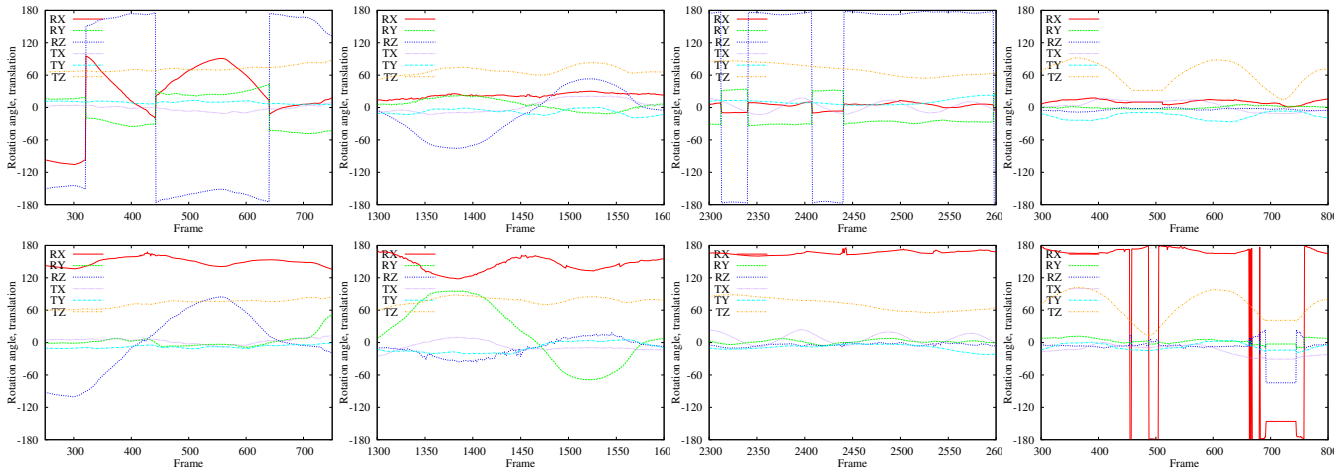


Figure 7. Graphs of estimated camera rotation (RX, RY, RZ) and translation (TX, TY, TZ) for representative videos. Smooth curves mean an error-free pose estimation; noise and "dents" in the curves indicate inaccuracies. $1^{\text{st}}$ row UMF, $2^{\text{nd}}$ row ALVAR. From left to right: perspective, occlusion, zig-zag, near-far. Major steps in the graph are caused by the gimbal lock and an angular over/underflow at $-\pi$, $\pi$.

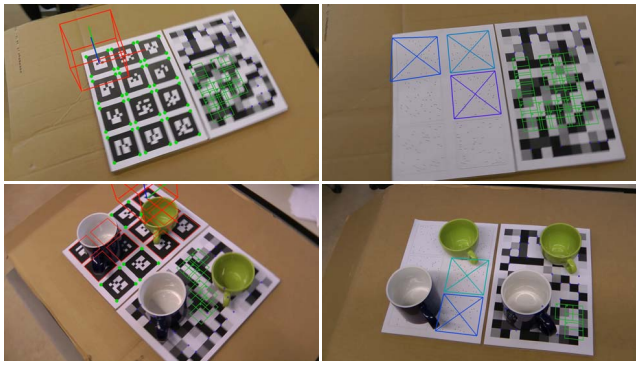

Figure 6. Illustrative frames of the side-by-side comparison video. **left:** Greyscale UMF vs. ALVAR. **right:** UMF vs. RDM. **top:** Motion blur tests. **bottom:** Occlusion tests. Videos were recorded in 1080p, capturing different classes of movement: zig-zag movement, upright rotation, rotation with severe perspective distortion, near/far movement, variable lighting conditions with fixed camera and general movement with occlusion.

| Method: | RDM | ALVAR | UMF |
|---|---|---|---|
| Average position variance: | 8.5 cm | 3.48 cm | 3.28 cm |
| Average rotation variance: | 0.049 | 0.035 | 0.024 |

Table 1. The average variance in position and rotation change using 10 frames for averaging in a 1080p 50FPS video. The rotation variance is expressed as variance of quaternions, since the euler angles are unstable due to the gimbal lock. (Note: RDM gave highly unstable results and the low average variance in rotation is caused mainly by the low detection rate. For the *rotation* test video it gave 0.080 variance.)

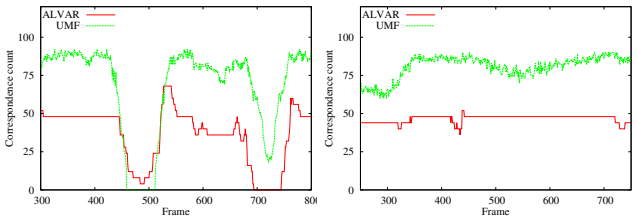


Figure 8. Number of correspondences used by ALVAR (red) and UMF (green) for the camera pose estimation. **left:** near-far video, **right:** perspective (refer to Fig. 7). More points naturally mean a more stable and precise camera pose estimation and better tolerance to wrongly detected points (caused by a motion blur, partial occlusion, etc.).

precision of our algorithm we used the local variance (in time domain) of the estimated camera pose (position and rotation) – see Tab. 1. Low local variance means that the results of camera localization are smooth. We did not include RDM, since its stability was notably worse (Tab. 2) and ALVAR thus serves as a good reference for precision evaluation. Our method gave smoother results thanks to the good spatial distribution of matched points between 2D and 3D and ALVAR's inability to find the corners of the individual markers precisely in blurred images and for partially occluded corners. The number of detected corners used for the camera pose estimation is shown in Fig. 8.

Apart from the precision we also show in Table 2 the success rate for each method in every category of videos. Random dot markers were the least successful, mostly due to their high sensitivity to the motion blur. But even for a fixed camera or rotation with minimal blurring it gave the worst results. ALVAR and UMF were both very successful and gave very similar results. The only major difference was for zooming and occlusion. Figure 6 shows the clear advantage of our continuous marker field over ALVAR. ALVAR only detected two completely visible markers and one which had one edge slightly occluded. On the contrary, our method was able to detect sub-markers and corner points even between the cups.

| Method | RDM | ALVAR | UMF |
|---|---|---|---|
| Lighting | 89.7 | 100.0 | 100.0 |
| Perspective | 42.7 | 100.0 | 100.0 |
| Near/Far | 75.8 | 91.3 | 93.4, 94.6 |
| Rotate | 94.7 | 100.0 | 100.0 |
| Zig-Zag | 29.6 | 98.3 | 97.5, 97.4 |
| Occlusion | 38.5 | 93.0 | 94.0, 96.5 |
| Overall | 61.8 | 97.1 | 97.8 |

Table 2. Marker field detection success rates in %. For UMF, rates from comparison videos with RDM and ALVAR are given separately. Success rate is the fraction of video frames where at least one of the markers was correctly detected in all the video frames.

### 4.2. Pixel Footprint and Computation Complexity

Table 3 shows the speed of the three tested algorithms and the breakdown of speed of our marker detection algorithm. Our algorithm was more than $3\times$ faster than ALVAR and visited on average about $5.3\,\%$ of all pixel points. A small memory footprint is an important property for ultra-mobile processors where the memory accesses are slow due to limited caching, etc.

| RDM | ALVAR | UMF | (edge | grid | match | cam | sref) |
|---|---|---|---|---|---|---|---|
| 164.4 | 30.1 | 8.8 | (3.8 | 1.1 | 0.3 | 0.7 | 2.9) |

Table 3. Breakdown of speed in milliseconds for 1080p videos using a mid-range Intel(R) Core(TM) i5 CPU 661 (3.33GHz) CPU. **edge:** edgel detection in scanlines (Sec. 3.1.1); **grid:** reconstructing the grid using RANSAC and vanishing point detection (Sec. 3.1.2 and 3); **match:** edge direction detection and position decision making (Sec. 3.2 and 3.3); **cam:** camera pose estimation based on the found matches; **sref:** processing in subwindows and position refinement by iterative search for more corner points (Sec. 3.4).

## 5. Targeted Applications

Here we give three targeted applications that guided us towards the development of the marker-based camera localization. While camera localization in natural scenes (SLAM/PTAM) is already achieving very good results and some applications do not require markers anymore, these sample applications deal with scenes where presence of reliable natural keypoints is impossible or undesirable.

### 5.1. Screen-to-Screen Task Migration

Along with cloud computing, a direct visual interaction between desktop and ultramobile devices is of interest [4], [19]. When the screen does not contain enough unique keypoints (often!), our marker field ensures the localization (Fig. 9). We are experimenting with possibilities of mixing the marker field in an unobtrusive way into any – static or dynamic – on-screen situation.

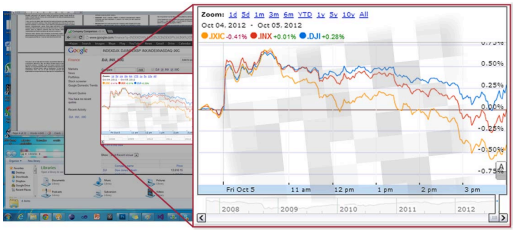

Figure 9. On a large desktop screen, a marker is mixed into the image for short periods of time so that a mobile device can reliably capture the exact location within the screen. Once the mobile device knows the location, the marker is displayed only in the vicinity of the mobile's view frustum so that it is as unobtrusive as possible. If enough distinct and stable keypoints are present, the marker is completely hidden and camera the pose is tracked.

### 5.2. Effortless Chromakeying

Chroma keying [7] is one of the widely used techniques in film production which is used to replace constant color with another scene reflecting the camera movement. The

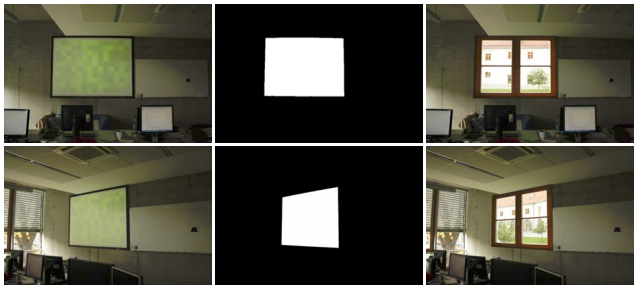

Figure 10. Matchmoving by the Uniform Marker Field. **left:** Image captured by the camera. **middle:** Alpha matte. **right:** Composite image with 3D scene rendered to match the camera pose.

camera pose can be determined by using sensors mounted to the camera (e.g. Insight VCS1[1]) or by camera rigs that can be programmed to follow a pre-defined track (e.g. Cyclops or Milo control rigs from Mark Roberts Motion Control[2] or TechnoDolly[3]). Camera movement recovery is a technique which estimates the movement using markers or keypoints placed and detected on the mating plate[4]. This process, also called matchmoving, often involves a considerable amount of manual work in order to match and annotate the markers. The marker field presented in this work can be used for the camera pose estimation without any human effort involved in the tracking (Fig. 10).

### 5.3. Tabletop Scene Interaction

One strong application of near-eye see-through glasses (recently becoming generally available) is augmenting interaction in tabletop scenarios [12]. Tracking of keypoints

[1] http://www.naturalpoint.com/optitrack/products/insight-vcs/
[2] http://www.mrmoco.com
[3] http://www.supertechno.com/product/technodolly.html
[4] http://www.fxguide.com/ – Art of Tracking

can be used for the camera pose estimation (e.g. [10]), but the presence of a visually unobtrusive and cheaply detectable marker field can provide a reliable starting point for the tracking and offload some of the expensive computation (Fig. 11). p

Figure 11. Tabletop interaction example. The marker field covers the whole table surface and fosters the camera pose estimation.

## 6. Conclusions

We presented a new design of marker fields whose square modules are greyscale and the location within the field is determined by the edges' gradients. Then, we proposed an efficient and reliable algorithm for detection of the marker field. We discussed three representative target applications where planar marker fields are desirable.

The results confirm that marker fields based on edges between the greyscale modules outperform the existing comparable solutions: arrays of black-and-white markers and random dot fields. The detection algorithm is efficient and reliable – because the grid of squares is being detected as a whole and the edgels thus can be detected roughly and sparsely. The detection rates and accuracy are about the same as for state-of-the-art algorithms (represented by ALVAR) or better. However, our detector is more than $3\times$ faster and it visits only a small fraction of image pixels ($\sim 5\,\%$). This opens space for implementation for ultra-mobile devices and specialized embedded sensors.

We are working on an altered algorithm that will not require the marker to be planar – on the contrary, the marker could be strongly deformed as on a cloth or wrinkled paper. The omnipresence of the detectable edges in the marker field will allow for real-time and precise detection of a deformed marker field. We will further experiment with the color marker fields where shades of grey are replaced by different tones of color. The abundance of localization information will allow for introducing further constraints into the marker field design – namely similarity to a given raster image. We expect these markers to be found even more aesthetically pleasing to the user.

Please, refer to the supplementary video for sample videos and more detailed comparison of the evaluated algorithms.

## Acknowledgements

This research was supported by the research project CEZMSMT, MSM0021630528, by the CEZMSMT project IT4I - CZ 1.05/1.1.00/02.0070, and by project V3C, TE01020415.

## References

[1] ALVAR tracking subroutines library web page. http://www.vtt.fi/multimedia/alvar.html.

[2] B. Atcheson, F. Heide, and W. Heidrich. CALTag: High precision fiducial markers for camera calibration. In *Proc. VMV*, 2010.

[3] J. Burns and C. J. Mitchell. Coding schemes for two-dimensional position sensing. *Institute of Mathematics and Its Applications Conference Series*, 45:31, 1993.

[4] T.-H. Chang and Y. Li. Deep shot: a framework for migrating tasks across devices using mobile phone cameras. In *Proc. SIGCHI*, 2011.

[5] M. Fiala. ARTag, a fiducial marker system using digital techniques. In *Proc. CVPR*, 2005.

[6] M. Fiala. Designing highly reliable fiducial markers. *IEEE T. Pattern Anal. Mach. Intell.*, 32:1317–1324, July 2010.

[7] J. Foster. *The Green Screen Handbook: Real-World Production Techniques*. Number v. 978, nos. 0-52106 in The Green Screen Handbook: Real-world Production Techniques. John Wiley & Sons, 2010.

[8] M. Hirzer. Marker detection for augmented reality applications. Technical report, Inst. for Comp. Graphics and Vision, Graz Univ. of Tech., AT, 2008.

[9] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based ar conferencing system. In *Proc. IWAR*, 1999.

[10] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. ISMAR*, 2007.

[11] C. Pirchheim and G. Reitmayr. Homography-based planar mapping and tracking for mobile phones. In *ISMAR*, 2011.

[12] O. B. R. Raskar. *Spatial Augmented Reality: Merging Real and Virtual Worlds*. A K Peters/CRC Press, 2005.

[13] F. Schaffalitzky and A. Zisserman. Planar grouping for automatic detection of vanishing lines and points. *Image and Vision Computing*, 18:647–658, 2000.

[14] I. Szentandrási, M. Zachariáš, J. Havel, A. Herout, M. Dubská, and R. Kajan. Uniform Marker Fields: Camera loc. by orientable De Bruijn tori. In *ISMAR*, 2012.

[15] H. Uchiyama and E. Marchand. Deformable random dot markers. *Proc. ISMAR*, pages 237–238, 2011.

[16] H. Uchiyama and E. Marchand. Toward augmenting everything: Detecting and tracking geometrical features on planar objects. In *Proc. ISMAR*, 2011.

[17] H. Uchiyama and H. Saito. Random dot markers. In *IEEE Virtual Reality Conf. (VR)*, 2011.

[18] A. Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.

[19] G. Woo, A. Lippman, and R. Raskar. VRCodes: Unobtrusive and active visual codes for interaction by exploiting rolling shutter. In *Proc. ISMAR*, 2012.