Gravity-Aware Handheld Augmented Reality

Daniel Kurz* Selim Benhimane†

metaio GmbH

Figure 1: Gravity-awareness in handheld AR: detecting vertical surfaces, such as building façades, improves when using gravity-aligned feature descriptors (GAFD) [12] (left). For horizontal surfaces, such as a magazine on a table (center), we introduce gravity-rectified feature descriptors (GREFD) that describe a feature based on a gravity-rectified camera image. We also show, how inertial sensors enable full 6 DoF pose estimation from horizontal surfaces with an occlusion-invariant edge-based detection method that only supports similarity transforms (right).

Abstract

This paper investigates how different stages in handheld Augmented Reality (AR) applications can benefit from knowing the direction of the gravity measured with inertial sensors. It presents approaches to improve the description and matching of feature points, detection and tracking of planar templates, and the visual quality of the rendering of virtual 3D objects by incorporating the gravity vector. In handheld AR, both the camera and the display are located in the user's hand and therefore can be freely moved. The pose of the camera is generally determined with respect to piecewise planar objects that have a known static orientation with respect to gravity.

In the presence of (close to) vertical surfaces, we show how gravity-aligned feature descriptors (GAFD) improve the initialization of tracking algorithms relying on feature point descriptor-based approaches in terms of quality and performance. For (close to) horizontal surfaces, we propose to use the gravity vector to rectify the camera image and detect and describe features in the rectified image. The resulting gravity-rectified feature descriptors (GREFD) provide an improved precision-recall characteristic and enable faster initialization, in particular under steep viewing angles. Gravity-rectified camera images also allow for real-time 6 DoF pose estimation using an edge-based object detection algorithm handling only 4 DoF similarity transforms. Finally, the rendering of virtual 3D objects can be made more realistic and plausible by taking into account the orientation of the gravitational force in addition to the relative pose between the handheld device and a real object.

1 Introduction

Recently, handheld Augmented Reality (AR) entered the mass market. The fundamental enablers are modern mobile phones and tablets that conveniently combine a camera, different positioning and orientation sensors, memory, fast processors, GPUs and a display in virtually everybody's pocket or handbag. In particular AR browsers, such as junaio[1], are becoming increasingly widespread. Among other features, they offer location-based services where points of interest (POIs) are displayed overlaid on the live camera feed of the device's back facing camera. The position and orientation of the device in the world is determined based on GPS, a digital compass and inertial sensors. While inertial sensors deliver quite accurate measures of the gravitational force, digital compasses and GPS are inaccurate and highly dependent on the surrounding environment. The precision might be sufficient for displaying POIs that are far away, but when it comes to rendering virtual 3D models precisely registered with the real environment, e.g. in the scenario shown in figure 1 on the left, it would not be possible to use pure sensor-based tracking. The only way to achieve precisely registered augmentations in handheld AR is to use computer vision algorithms. They enable to determine the position and orientation of the camera with respect to a known environment based on information from the camera image. Such techniques are known to be computationally expensive, particularly on mobile devices, and the state-of-the-art is still far away from being able to localize oneself at any place on earth using computer vision.

This paper investigates how inertial sensors that measure the gravitational force can support vision-based handheld AR. We therefore take a closer look at the needs and constraints of real world handheld AR applications and present means to improve three fundamental parts in the pipeline of video-see-through AR. These are feature (or keypoint) description and matching, (edge-based) template detection and tracking as well as the integration of virtual 3D objects into the live camera stream. While the latter is tightly coupled to video-see-through AR, the others are fundamental tasks in computer vision and might also find applications in other research areas using computer vision.

Classical desktop and kiosk video-see-through AR applications are based on tracking the position and orientation of a moving object, e.g. a cardboard box of a product, with a static camera and

*e-mail: Daniel.Kurz@metaio.com
†e-mail: Selim.Benhimane@metaio.com

[1] http://www.junaio.com

display. A fundamental finding when looking at real world handheld AR applications is that here the situation is the opposite. While camera and display are in the user's hand and therefore can move, the surfaces that are used for tracking are very often static. Furthermore, we observed that many of the used static surfaces are either (close to) vertical, e.g. posters, paintings, displays and building façades, or (close to) horizontal, e.g. a magazine on the table or a print on the floor.

1.1 Outline and contributions

After discussing the related work in section 2, we will shortly explain the hardware and calibrations used for our evaluations. We then present means to improve the description and matching of feature points given inertial sensor measures in section 4. We show that gravity-aligned feature descriptors (GAFD) [12] can be applied to improve tracking (re-)initialization in handheld AR applications dealing with vertical surfaces. For applications tracking a horizontal planar surface, such as the magazine on a table in figure 1, we introduce gravity-rectified feature descriptors (GREFD) that use the measured gravity vector to rectify the camera image before feature point detection and description. Section 5 investigates how this concept can also be applied to different template tracking algorithms on horizontal surfaces. We propose to perform 4DoF similarity transform image registration methods on "gravity-rectified" camera images and show how this enables 6 degrees of freedom (DoF) camera pose tracking. Finally, we present an application of the measured gravity to increase plausibility and realism of virtual 3D objects that are used to augment arbitrarily orientated surfaces in section 6. Section 7 concludes the paper and discusses future work.

2 Related work

Handheld AR is a comparatively young field of research. Early approaches were using fiducials to track the camera pose either in an outside-in manner, i.e. an external camera is tracking the position of the handheld device from fiducials attached to the device, or with a marker-based inside-out tracker using the camera attached to the handheld device [26]. Klein *et al.* [9] proposed a combined approach using an outside-in tracker based on LEDs combined with inside-out tracking of natural features on a tablet PC. Wagner *et al.* [25] presented a full 6 DoF (inside-out) natural feature tracking framework running on a mobile phone. The two algorithms used were simplified and optimized versions of SIFT [16] and FERNS [20] which are purely vision-based and do not use any sensor information. In this paper, we show how vision-based algorithms can benefit from gravity measurements.

Other approaches to camera pose tracking for handheld AR are based on edges. In [7] a descriptor is built for each concavity in a shape to track from. This makes it hard to detect and track shapes with repetitive concavities, as they will have similar descriptors. Also, the descriptor itself is not robust against occlusions. In section 5.2, we present an inertial sensor-supported edge-based tracking algorithm which deals with both repetitive structures and occlusions.

Lee *et al.* [13] recently presented an approach inspired by Gepard [8] to track vertical and horizontal patches on a mobile phone. While they make use of the accelerometers in the phone to rectify reference patches, the sensors are later on not used for detection and tracking.

We recently presented gravity-aligned feature descriptors (GAFD) [12]. We showed that aligning local feature descriptors, such as SIFT, with the gravity instead of gaining a canonical feature orientation from the intensities of neighboring pixels increases descriptor distinctiveness. The impact is particularly high for congruent and near-congruent features in different orientations which are widespread in urban and man-made environments.

To compensate for the drift of a gyroscope used to track the orientation of an HMD, Satoh *et al.* [22] use simple visual tracking of landmarks. The dissimilarity they use is the sum of absolute image intensity differences of the pixels around a feature. In order to predict the position and orientation of features in a new frame, they use the relative orientation measured with the gyroscope. While this makes their feature descriptor invariant to in-plane rotations, it does not support changes in scale or perspective transformations. This clearly limits it to applications where the user does not change its position. Also, it is not robust against illumination changes which is critical for outdoor applications. A similar approach warps the entire image based on relative changes in orientation measured with a gyroscope to support tracking of features from frame to frame [6]. In contrast, GAFD is invariant to scale and rotation and robust against changes in perspective and illumination as SIFT with an improved precision-recall characteristic.

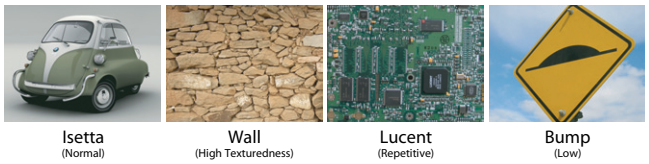

Figure 2: The templates used through this paper, as used in [14].

There are different model-based tracking algorithms that use a gyroscope to measure the relative change in orientation between two frames to predict the new pose. The model is then rendered using the predicted pose. The tracking can be either edge-based [21] or feature-based [4] and tracks the transformation between the rendered image of the model and the real image. This kind of combination with inertial sensors improves vision-based algorithms particularly for fast camera movements.

Lobo and Dias [15] show that different challenges in computer vision can be simplified when using the measured gravity as a vertical reference. They show that given the direction of the gravity, the horizon line can be determined in the camera image and given only one vanishing point in addition enables estimation of the camera's focal length. Attaching inertial sensors to a stereo camera, enables them to segment level planes and reconstruct vertical features.

Kotake *et al.* presented methods to simplify pose estimation from feature correspondences using inclination constrains provided by an accelerometer [10]. They also show how the complexity of pose estimation from edge-correspondences can be improved using inclination constrains [11]. While they improve pose estimation given constraints and feature correspondences, we show means to improve the quality of the feature correspondences using GAFD and GREFD. In other words, they reduce the complexity of solving a task, while we reduce the complexity of the task itself.

3 Hardware and calibration

In this paper, we use different mobile devices for evaluations. For all of them, the intrinsic camera parameters were calibrated offline. In addition, the transformation between the coordinate systems of the camera and the inertial sensors which are rotations about multiples of 90 degrees have been determined according to the devices' documentation. While accelerometers do not only measure the gravity, but all accelerations applied to the device, the gravity vectors we use are provided by Apple's Core Motion Framework. They combine the acceleration measures with gyroscope data to eliminate the impact of user-applied acceleration to the device. For a fair comparison of algorithms, we made sure they work on comparable data. Therefore, we capture image sequences with corresponding gravity vectors. To enable a constant capturing frame rate of around 25 Hz, all image and inertial sensor data is accumulated in main memory while capturing and afterwards written into files. The different algorithms are then evaluated offline on a PC.

4 GRAVITY-AWARE FEATURE DESCRIPTION AND MATCHING

Many tasks in computer vision require the discovery of points corresponding to the same physical 3D point in different camera images. One example used in AR is finding an initial estimate of the camera pose when there is no knowledge of the camera pose in the preceding frame. A common approach, as described in [16], is to first detect features (or keypoints) in each image at positions that are expected to have a high repeatability. To enable comparison and matching of features, a description of each feature is needed. A repeatable scale and an orientation are computed for each feature based on local image intensities. By transforming the coordinates of the descriptor relative to this orientation and scale, the descriptor becomes invariant to scale and rotation.

The two important characteristics of feature descriptors are distinctiveness and invariance to changes in viewpoints and illumination. While features corresponding to the same physical point should be described with similar descriptors under different circumstances, features corresponding to different physical points in the scene should be described with a distant descriptor with respect to a certain distance measure.

4.1 Gravity-aligned descriptors on vertical surfaces

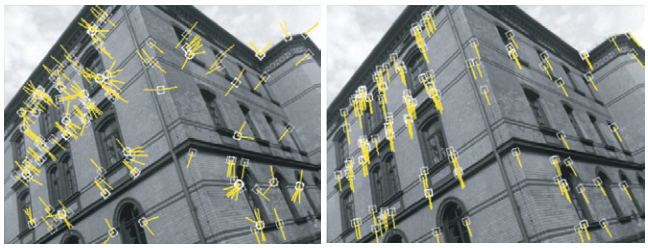

Figure 3: While regular approaches compute the feature orientation (illustrated in yellow) based on image intensities (left), we propose to use gravity-aligned feature descriptors (right).

We recently showed how inertial sensors can both improve and speed up the process of feature description and matching, by aligning the orientation of feature descriptors, e.g. SIFT [16], with the direction of the measured gravity. Gravity-aligned feature descriptors (GAFD) [12] can be applied in any application dealing with static and vertical or close to vertical surfaces, such as the building façade in figure 3. In particular, in the presence of congruent or near-congruent features in different orientations, the recall-precision characteristic of GAFD clearly outperforms regular feature descriptors that gain a canonical feature orientation based on the intensities of pixels in a region around it. While this makes regular feature descriptors invariant to rotation, it results in indistinguishable descriptors for congruent features in different orientations. As GAFD uses the direction of the gravity instead, it describes such features in a distinctive way.

Besides the improved quality of feature matches, matching can also be significantly sped up when computing the canonical (local) orientation of each feature as in standard approaches and then computing the relative local orientation of a feature as the difference between the local and the global orientation of the gravity. This relative orientation can be used as an additional part of the descriptor, as it is static. We showed that by only comparing features with similar relative local orientation, about 85% of descriptor comparisons can be saved while the results are still clearly superior compared to regular feature descriptors.

In our recent work [12], we showed that using GAFD improves recall-precision of SIFT [16] for different template images. It has also been shown to increase recognition rates in a museum guide on a mobile phone that recognizes artworks. Here, we will focus on how it affects feature-based localization of a template, which is crucial to initialize camera pose tracking for handheld video-see-through AR applications. Many such applications rely on pose estimation from features on vertical static surfaces. Examples include posters or billboards, cars, and increasingly more important building façades for large scale self localization in urban outdoor environments.

To ensure a fair comparison, we recorded image sequences with the measured gravity vector for each image and then run the evaluations offline on a PC. Nevertheless, the detection algorithm we use for evaluation is optimized to run on mobile devices in real-time. In an offline step, the algorithm is provided with a reference image of the template to detect. We use a fronto-parallel photo of the template captured with the same camera that took the image sequences. From this reference image, we extract up to 200 features and describe them with reference feature descriptors. For each current image from a sequence, we first detect and describe current features. The next stage then finds the nearest neighbor (with the closest descriptor) of each reference feature in the set of current features using exhaustive search. This results in a set of 2D-2D correspondences, of which we only keep those where the ratio of distances of the best and the second best match is above a particular threshold. The remaining correspondences are used by PROSAC [5] to find a homography that maps the current image to the reference image. If such homography could be found, we eventually refine it using Inverse Compositional [1] image registration. This homography, together with the intrinsic parameters of the camera, can then be used to compute the pose of the camera to enable rendering registered 3D models into the camera image.

In our evaluation, we compare two versions of the localization algorithm that only differ in the feature orientation assignment step. While the "regular" algorithm uses up to 6 directions of dominant gradients in a region around the feature as orientation and computes one descriptor for each orientation, the "GAFD" algorithm computes the orientation as the projection of the gravity vector onto the image plane. Note, that the direction of the gravity is not necessarily uniform across the camera image, as can be seen in figure 3 on the right. Given the normalized gravity vector $\mathbf{g} = [g_x,\, g_y,\, g_z]^\top$, where $\|\mathbf{g}\| = 1$, and the (3×3) intrinsic matrix $\mathbf{K}$ of the camera, the projected 2D gravity direction $\mathbf{d} = [d_u,\, d_v,\, 0]^\top$ at a pixel $\mathbf{p} = [u,\, v,\, 1]^\top$ is computed as

$$\mathbf{d} = \mathbf{p}' - \mathbf{p}\,,$$

where $\mathbf{p}' = [u',\, v',\, 1]^\top$ can be computed from

$$[wu',\, wv',\, w]^\top = \mathbf{p} + \mathbf{K}\mathbf{g}\,.$$

The fact that the lengths of $\mathbf{d}$ and $\mathbf{g}$ are arbitrary lead to a computationally cheap formulation of the direction as

$$\mathbf{d} \propto \left[g_z(p_u - u) + f_u g_x,\, g_z(p_v - v) + f_v g_y,\, 0\right]^\top,$$

where $[p_u,\, p_v]^\top$ is the principal point and f_u and f_v are the horizontal and vertical focal lengths of the camera. The orientation angle of a feature is eventually computed as the arctangent of (d_v/d_u).

After quantization of the angle, the descriptor uses the assigned orientation to perform an in-plane rotation of its coordinates. Finally, a 48-dimensional vector is computed based on gradient histograms which will be referred to as the feature descriptor. The similarity measure used to compare feature descriptors is the Euclidean distance. We judge the properness of a detection and the computed homography by computing the Zero-mean Normalized Cross Correlation (ZNCC) of the reference image and the current image warped with the homography. Our assumption is that with an increasing ZNCC value, the accurateness of the homography increases. If the algorithm does not find a homography the ZNCC is set to -1.0.

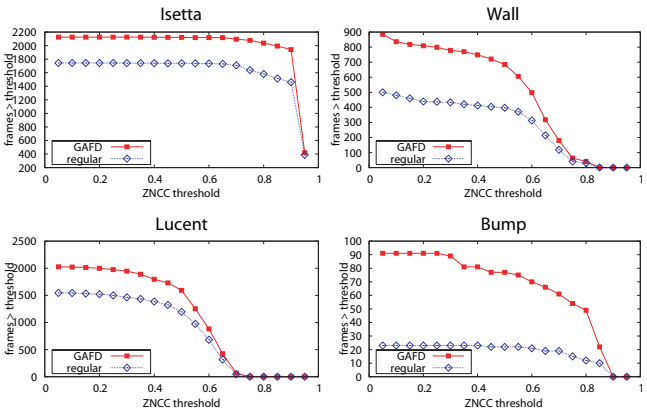

Figure 4: The plots show the number of images, where a localization could be performed that results in a ZNCC greater than a particular threshold as a function of the threshold used. GAFD provide better results for all templates in figure 2 than regular feature descriptors.

In order to cover a wide range of objects to track, we chose one template of each of the four groups in [14], cf. figure 2. We attached a print-out of each template of the size (160×120) mm to a vertical surface in an upright orientation using a water-level. For each of the four reference templates, we record a sequence of 480 camera frames with an iPhone4 at a resolution of (480×360) pixels and store the measured gravity for each frame. As reference image, we take one fronto-parallel image of the template with the same camera. The reference images have a resolution of (320×240) pixels which approximately matches the average size of the template in the image sequences.

To reduce the effect of the randomization in PROSAC [5], we evaluated each sequence five times, resulting in 2400 ZNCC computations in total. The results in figure 4 show the number of images, where a localization could be performed that results in a ZNCC greater than a particular threshold for different thresholds between 0 and 1. Even though the localization performance differs significantly among the different templates, the average ZNCC improved clearly when using GAFD for all of them. On average, over all templates and all thresholds plotted in figure 4, the number of correctly localized images is slightly more than doubled when using GAFD compared to regular descriptors.

As one important application of GAFD is tracking initialization in urban outdoor environments, we also carried out the same experiments explained above on image sequences of real building façades. For each sequence of 480 images, a reference image has been manually rectified and extracted from an additional image. The façades and the areas used as reference are illustrated in figure 5 with a yellow frame. Again, each image has been evaluated five times to reduce randomization effects. The results are shown in figure 5. While the improvements have a high variance over the different sequences, our approach outperforms regular feature descriptors in all cases. The average increase in detection for these sequences of over 50% clearly shows the potential and applicability of GAFD for outdoor tracking, which we consider one of the most exciting areas in the near future of handheld AR.

4.2 Features on horizontal surfaces

Another important category of handheld AR applications is based on tracking the camera pose with respect to a planar object on a horizontal plane. Examples include games or marketing applications that track books, magazines, menus in a restaurant, prints on the floor or product packages on a table. This section describes how the description of features can be improved knowing that the object

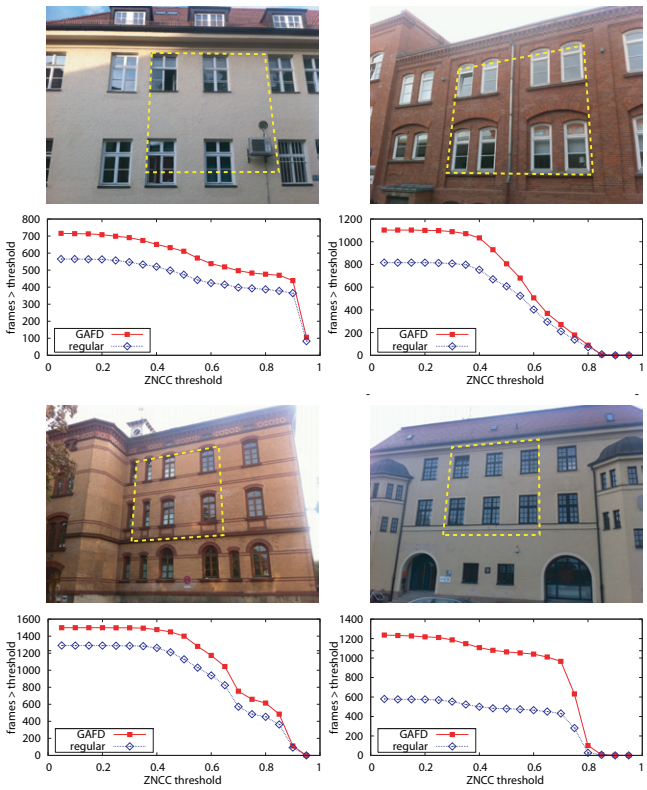

Figure 5: The four building façades used in the evaluation provide a good amount of repetitive features. The yellow frame indicates the individual areas used as reference template for feature-based localization. The corresponding plots show the number of images for which the algorithm could provide a transformation that results in a ZNCC greater than a particular threshold as a function of the threshold used. We clearly see that, with respect to building façades in urban outdoor environments, for any ZNCC threshold that the algorithm would use to decide about a correct localization, GAFD outperform regular feature descriptors.

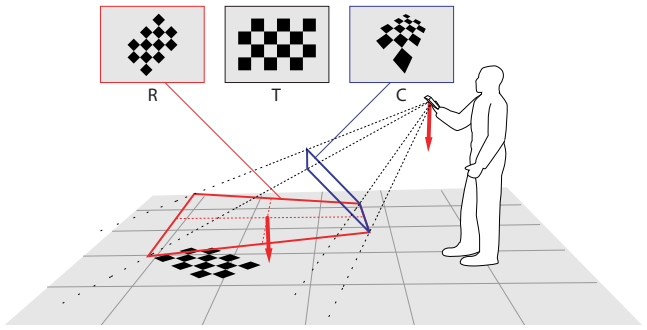

Figure 6: By projecting the camera image C onto a virtual image plane perpendicular to the measured gravity, we gain a synthetic gravity-rectified view R of a template T which is located on a (close to) horizontal plane.

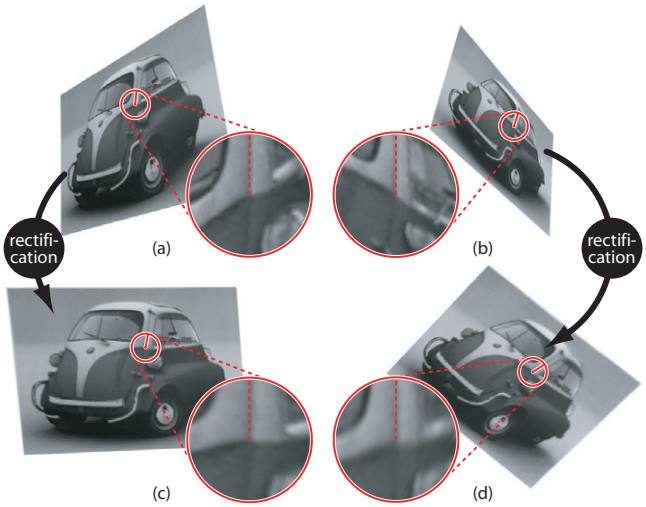

Figure 7: The pixel intensities in the region around features corresponding to the same physical 3d point on a horizontal surface become more similar (c,d) after gravity-rectification of the camera image compared to the original images (a,b).

they correspond to is located on a horizontal plane and having a measure of the direction of the gravity. Note, that in this case the gravity vector is the normal of the plane the object is located on.

We introduce a virtual image plane which is perpendicular to the measured gravity vector as illustrated in figure 6. Warping a camera image C onto this virtual image plane results in a gravity-rectified image R that appears as if it was taken in a camera orientation where the optical axis is aligned with the gravity vector. While the original image C shows perspective distortions with respect to the template (reference) image T, the gravity-rectified image R does not. A very similar approach is used in [13] to rectify reference templates located on horizontal surfaces offline before tracking.

In our implementation, the gravity image rectification is done with inverse warping in order to avoid having holes in the gravity-rectified image. Therefore, we compute a matrix $\mathbf{H}$ that aligns the gravity vector $\mathbf{g}$ with the optical axis $\mathbf{z} = [0,\ 0,\ 1]^\top$.

We assume the z-component of the gravity vector to be non-zero. The case $g_z = 0$ happens only when the optical axis is perpendicular to the gravity vector. Since we assume that the tracked objects are on (close to) horizontal planes, the assumption $g_z \neq 0$ is generally valid when the tracked objects are in the camera field of view. Now, let $\mathbf{g}_1$ and $\mathbf{g}_2$ be two vectors defining a plane with a normal $\mathbf{g}$.

These two vectors can be defined as follows:

$$\mathbf{g}_1 = [-g_z,\ 0,\ g_x]^\top \text{ and } \mathbf{g}_2 = \mathbf{g} \times \mathbf{g}_1.$$

The (3×3) matrix $\mathbf{H}$ defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}^{-1}$$

transforms the gravity vector $\mathbf{g}$ into a vector aligned with $\mathbf{z}$.

$$\begin{aligned} \mathbf{H}^{-\top}\mathbf{g} &= \left(\begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}^{-1} \right)^{-\top} \mathbf{g} \\ &= \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}^{\top} \mathbf{g} \\ &= [0,\ 0,\ g_z]^\top = g_z \mathbf{z} \\ &\propto [0,\ 0,\ 1]^\top = \mathbf{z} \end{aligned}$$

Given the matrix of intrinsic parameters $\mathbf{K}$, transforming every captured image using an inverse warping with the homography $\mathbf{G}$ defined as

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix} \mathbf{K}^{-1}$$

allows rectifying perspective distortions of any plane orthogonal to the gravity vector. This means that when a tracked object is lying on a horizontal surface, its fronto-parallel reference template is transformed in the gravity-rectified image with a similarity transformation. Our implementation uses $\mathbf{G}' = \mathbf{K}\begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \mathbf{z} \end{bmatrix}\mathbf{K}^{-1}$ instead, which provides the same property if the camera intrinsics do not contain any shear and pixels are quadratic, but in contrast to $\mathbf{G}$ results in an image in the same domain as the original one.

In the following, we evaluate the impact of this rectification to the recall-precision characteristic of a local feature descriptor. Inertial-sensor based rectification does not increase distinctiveness of congruent features in different orientations. The improvement of gravity-rectified feature descriptors (GREFD) over standard approaches is that, due to the rectification, the region around each feature point, which is used to compute its orientation and the actual descriptor, is more similar, even under steep viewing angles. Figures 7a and 7b show the region around features corresponding to the same physical point in two different views. When comparing this with the regions around the corresponding features in the rectified camera images (figures 7c and 7d), it is clearly visible that the regions are more similar in this case. Below, we will show that this also results in more similar descriptors which leads to an improved recall-precision characteristic. A similar effect is achieved in [27], where the normal of a surface is reconstructed from the local geometry in a range image instead of measuring gravity.

The feature descriptor used in this evaluation equals the regular feature descriptor in section 4.1 which is optimized to run in real-time on handheld devices. The orientation assignment is based on the dominant gradient directions. Figure 1 (center) illustrates the support regions used for orientation assignment and description of the features with a white quad in the central image. These regions correspond to squares in the gravity-rectified image and the red lines indicate the orientation of each feature.

As in 4.1, we use the four template images shown in figure 2. For each template, we took two sets of ten images at a resolution of (480×360) pixels each with an iPod touch 4G and additionally stored the gravity measurement for each image. While in the first set, the viewing axis of the camera was close to perpendicular to the template plane, the second sequence images the template under steeper angles. Our hypothesis is that the impact of inertial sensor-based rectification is bigger on the latter group of sets than the first group. Given the images and the corresponding gravity vectors, we synthesize gravity-rectified images R using inverse image warping, as explained above. As image warping is computationally expensive, we use both nearest neighbor (GREFD NN) and bilinear (GREFD BIL) interpolation and evaluate the impact of the technique used. Two results of the rectification process using bilinear interpolation are shown in figure 7. In addition to the image sets, we take one reference image per target as a fronto-parallel view.

From each camera image, we extract 200 features which are described with up to 6 descriptors depending on the number of dominant gradient directions. For each feature in a current camera image, a matching reference feature from the template is found using exhaustive nearest neighbor search. The distance measure used is the Euclidean distance of the two descriptors. Whether a match is correct or not is determined by transforming the position of the current feature into the coordinate system of the reference image using a homography computed based on the known positions of the four template corners in the current image. If the transformed position of the current feature and the position of the matched reference feature differ by less than 5 pixels, the match is considered correct.

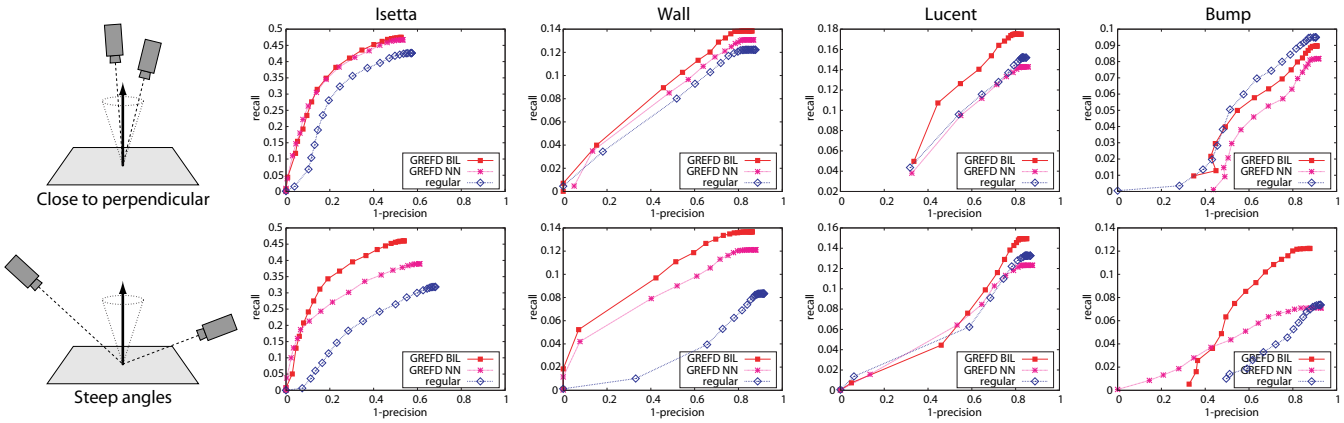

Figure 8: Precision-recall characteristic of feature descriptors for different target images. The upper row compares using the original image ("regular"), with gravity-rectified versions using nearest neighbor interpolation ("GREFD NN") and bilinear interpolation ("GREFD BIL") for images taken from close to perpendicular viewpoints. The lower row shows the same properties for images taken from steep angles. In general, gravity-rectification improves the characteristic when using nearest neighbor interpolation and even more significantly when using bilinear interpolation.

Similarly to Mikolajczyk and Schmidt [18], we compute *recall* vs. $1-precision$ for each image set to evaluate the different kinds of feature descriptors, where

$$recall = \frac{\#correct\ matches}{\#correspondences} \text{ and}$$

$$1-precision = \frac{\#false\ matches}{\#correct\ matches + \#false\ matches}.$$

The resulting precision-recall characteristic for the eight different image sequences is shown in figure 8. The samples for the plots have been created by using subsets of all matches whose Euclidean distance of the descriptors is below a particular threshold for different thresholds.

For the normally textured "Isetta" target and the highly textured "Wall", the results meet our expectations: rectification using nearest neighbor interpolation improves the matching quality and using bilinear interpolation leads to even bigger improvements. Also it is clearly visible that the impact is bigger under steep angles than for close to perpendicular camera poses. Interestingly, the characteristic of the description of features on the repetitive "Lucent" target does not considerably change when rectifying with nearest neighbor warping. One explanation for that is that the negative impact of fine details getting lost in nearest neighbor interpolation and the positive impact of rectification cancel each other out. When using bilinear interpolation, the recall is improved for nearly all samples, but for this target, the impact is clearly stronger for close to perpendicular viewpoints compared to steep angles. The "Bump" sign, with its very low texture, shows a similar behavior for steep camera angles as the "Isetta" and "Wall" targets. However, rectification downgrades the quality independent of the interpolation method used for close to perpendicular viewpoints.

Looking at the results above, we propose an adaptive approach that chooses the procedure based on the current inertial sensor measurements. If the camera is oriented in a close to perpendicular angle, we do not use any rectification, as we could not report on significant improvement that would be worth the extra effort of warping the image. For steeper angles up to 40 deg, the adaptive approach would use nearest neighbor interpolation, while for very steep angles, bilinear interpolation would be used to get the best matching quality possible.

It is important to keep in mind, that gravity-rectification may have side effects on the extraction of feature points. On the one hand, warping introduces new edges to the gravity-rectified image at the borders of the original camera image. However, feature point detection at these edges can be easily avoided, as their positions are known. On the other hand, it might happen, in particular under steep angles, that relevant parts of the camera image project beyond the dimensions of the gravity-rectified image. While the test images used in this evaluation where taken such that the template is always entirely visible in the gravity-rectified camera image, the system used in the following scales the warping based on the steepness of the camera. An empirically found method uses the square-root of the z-component of the normalized gravity vector, g_z, as scale.

$$\mathbf{H}_{scaled} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \sqrt{|g_z|}\mathbf{z} \end{bmatrix}^{-1}$$

Figure 10 compares in the left column the localization results of the "regular" template localization algorithm explained in section 4.1 with a version using GREFD as feature descriptor. The goal is to localize print-outs of the templates in figure 2, located on a horizontal surface. Results are rated by the ZNCC of the reference template and the current image warped with the estimated homography. The number of frames where a template could be localized with a ZNCC above a particular threshold increases for all templates and all threshold when using GREFD compared to regular descriptors.

5 Inertial Sensor-Aided Template Tracking

For arbitrary camera poses, the transformation an object undergoes when imaging it is a 3 DoF rotation, followed by a 3 DoF translation and eventually a projection onto the image plane. For planar objects, this perspective transformation can be fully described with a homography (which is a (3×3) matrix). This section deals with algorithms that can be used to track the position of a planar object in an image by estimating a homography which registers the current camera image with a template image of the planar object.

When the template is known to be located on a horizontal surface, we propose to rectify the camera image based on the measured gravity prior to aligning the camera image with the template. Figure 11a depicts a camera image and its gravity-rectified version in figure 11b. As can be seen, after rectification, the transformation between the template and the rectified image can be described with a similarity transform supporting 3 DoF translation and in-plane rotation only. Any image registration algorithm for similarity transforms can be applied to find this transformation. Having the similarity transformation, we propose to multiply it with $\mathbf{G}^{-1}$, which is the inverse of the transformation used to rectify the camera image. This results in the correct perspective transformation for registering

the current image with the template, cf. figure 11c. The motivation behind this procedure is, that tracking using similarity transforms is computationally less expensive and more stable than perspective tracking. In section 5.2 we will also show that it requires significantly less memory for particular algorithms, which is beneficial particularly on mobile devices.

5.1 Improving gradient descent image registration

A widely used class of image registration methods is based on gradient descent algorithms. These image registration methods are based on non-linear optimization algorithms that iteratively find a local minimum of a cost function which describes the difference between the template and the current image. The algorithms iteratively update the cost function and solve a system of equations to find the optimal parameters of some transformation that is applied to one of the two images.

Note that the implementation of a gradient descent image registration method supporting only similarity transformations is very inefficient because it does not enable linear parameterization. Instead, we use methods that support affine transformations, which additionally include shears and non-uniform scaling, but can be linearly parameterized with 6 parameters.

Given perfectly gravity-rectified images[2], this class of image registration methods improves in two ways. First, the optimization algorithm is much faster when using a gravity-rectified image as input. In fact, when the cost function needs to be updated at each iteration, an image warping using the transformation parameters is required. In case of using a gravity-rectified image, the registration method needs to estimate an affine transformation. Since affine warpings do not require divisions, the cost function update inside the optimization method is then much faster than when using generic perspective warpings based on standard homographies which require divisions. Also, since affine transformations have 6 parameters (degrees of freedom) to be estimated, solving the system of equations inside the optimization is much faster in the case of affine image registration compared to the standard perspective registrations which require 8 parameters to be estimated.

Second, in addition to the improvements in terms of computational speed, the optimization algorithm converges more often when we take the knowledge that a gravity-rectified image is given as input into account. Using an "affine" version of the optimization algorithms gives a better convergence rate compared to the standard "perspective" versions. We validated this using the same Matlab tool and the same reference (100×100) template as the ones used in [2]. This Matlab tool is available on the website of the Robotics Institute of the Carnegie Mellon University. In order to compute the convergence rate, we warp a reference template 9500 times using different random (but known) affine transformations. The affine transforms are computed by adding a Gaussian noise to the coordinates of 3 control points on the border of the square template (two corners of the template and the center of the other two corners). The standard deviation of the Gaussian noise is increased from 1 to 10 with steps of 0.5. We performed 500 random affine warpings for each standard deviation. The standard deviation of the Gaussian noise defines the amplitude of the affine warping.

On this data, we tried different registration methods namely the Forward Additive (FA) [17], the Forward Compositional (FC) [23], the Inverse Compositional (IC) [2] and the Efficient Second-Order Minimization (ESM) [3]. The affine and perspective version of each algorithm was used to register the template after each warping. In order to compute the convergence rate, we consider that an algorithm converged, if the spatial root-mean-square of the 3 control points is under 1 pixel after 15 iterations. For more information about the used benchmarking approach and the computation of the convergence rate, the reader can refer to [2]. Figure 9 plots the convergence rate for the different tested algorithms as a function of the amplitude of the warping.

[2]For now, we assume perfect gravity measures.

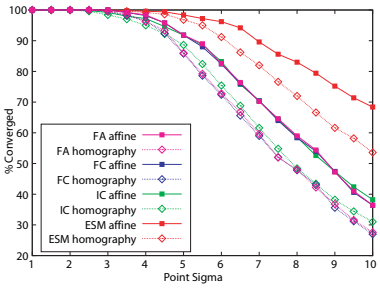

Figure 9: Comparison of the convergence rate of an affine template tracker with a perspective homography tracker on affine image transformations. As can be seen, the affine version converges more often than the one using a perspective homography for all tested methods.

It is possible to see that we get a higher convergence rate using the affine transform version of all tested algorithms and for all warping amplitudes than when using the version estimating a full perspective homography. As the amplitude of the warping increases, the rate of convergence of the methods that take into account the fact that the input image is gravity-rectified decreases slower than the rate of convergence of the methods that do not take the rectification into account. Similar results were obtained with all tested reference templates.

This means that performing a perfect gravity-rectification of the input image and using the affine version of the image registration improves not only the computational efficiency but also the convergence rate of their gradient descent algorithms.

The tracking system used for evaluation uses the localization algorithm explained in section 4.2 for initialization and the Inverse Compositional (IC) image registration method to track a template from frame to frame. If the IC tracker fails, i.e. the ZNCC between the reference image and the warped current image is below 0.7, the localization algorithm is used again to reinitialize.

Below, we will compare three different tracking systems. The "regular" system uses regular feature descriptors for initialization and IC template tracking that estimates a full homography (regular+homography). We compare this tracker with two versions using GREFD. While the first system uses an affine IC template tracker after GREFD-based initialization (GREFD+affine), the second uses an unchanged IC template tracker that finds a homography (GREFD+homography). The objective is to track (160×120) mm print-outs of the four template images shown in figure 2 that are located on a horizontal table. To ensure a fair comparison, we captured two images sequences and the corresponding gravity vectors as described in section 4.1 for each template image, and ran the tracking systems offline on a PC.

The resulting performance of the three different tracking systems is compared in figure 10 in the right column. The left column shows the detection results without IC frame-to-frame tracking for the same sequences as reference. The plots show the number of frames where the ZNCC was above a particular threshold as a function of the threshold. In all cases, the two trackers using the proposed gravity-rectified feature descriptors (GREFD) clearly outperform the regular tracking system. The question, if affine tracking of the gravity-rectified camera image improves tracking compared to a regular homography-based tracker does not have an universally valid answer. While for the "Isetta" and the "Wall" template, "GREFD+homography" performs clearly better than "GREFD+affine", it is not clear which system is better for the Lucent sequence. On the other hand, the "Bump" sequence clearly

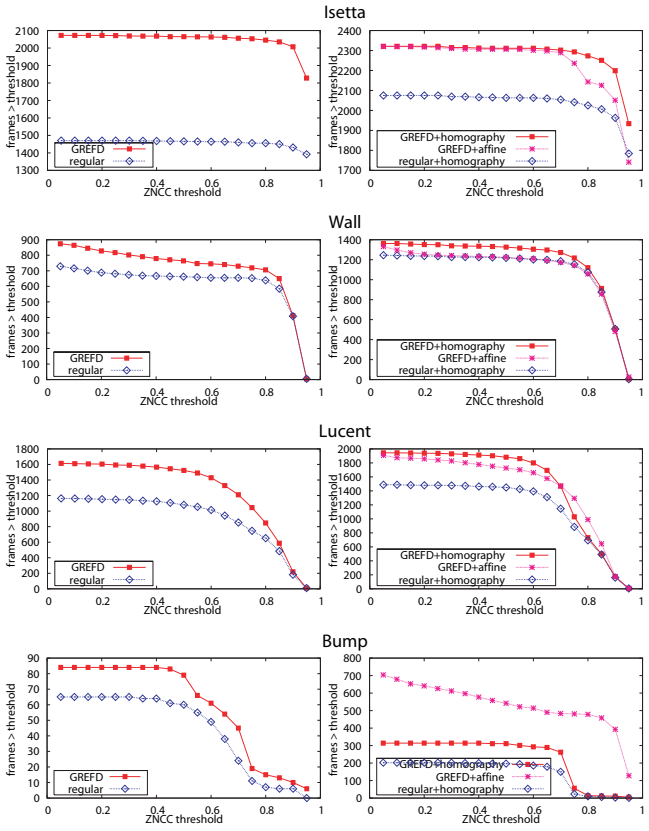

Figure 10: Number of frames where a transform was found resulting in a ZNCC greater than a threshold for a regular markerless tracking system and a modified version using gravity-rectified images. In all tests, gravity-rectification increases the number of successful frames.

performs better with the affine transform approach. The fact, that most real applications use template images that fall into a category with "Isetta" and "Wall", led to the conclusion, that in practice, and in the video footage available in the supplemental material, we use "GREFD+homography". While in theory affine tracking of a perfectly gravity-rectified image would outperform regular homography-based tracking as explained above, the gravity measures provided by current phones are in practice still too inaccurate and noisy. Together with slightly inaccurate camera intrinsics, this not only results in a jittery but also a not perfectly rectangular appearance of the template in the gravity-rectified camera images.

5.2 Enabling 6 DoF edge-based tracking

Template tracking, as described above, works well in a static and well-textured environment. However, in real world applications, parts of the template may be occluded in the current camera image, with the illumination changed non-uniformly compared to when the reference image has been taken or the template is hardly textured. In this case, edge-based approaches have shown to perform well. This section studies the how the concept explained above for gradient descent methods can be used with an edge-based method.

We investigate the applicability of the edge-based object detection algorithm by Steger [24] for handheld AR applications. The original algorithm handles similarity transforms only, i.e. translation, scaling and in-plane rotation. We present a simplified version that runs on mobile devices in real-time and show how an inertial sensor-based gravity-rectified camera image in combination with this approach enables estimating a full 6 DoF camera pose.

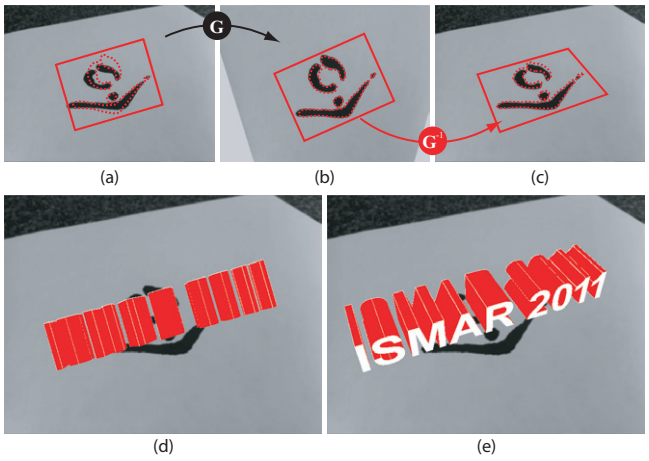

Figure 11: As soon as the camera's principal axis is not perpendicular to the tracked target, similarity tracking cannot estimate the transformation the template underwent in the camera image correctly (a,d). Using the proposed inertial sensor-based rectification of horizontal targets (b), a similarity tracker is able to estimate the transformation which results in a proper pose (c,e).

Given a reference image, we create synthetic views of the template under discrete samplings of all possible rotations and scalings. The step length used is 4 degrees for rotation and there are 20 supported scales uniformly distributed in the range $[0.6, 1.4]$. We use six pyramid levels with a scale factor of 1.5 between the individual levels to run the detection and tracking in a hierarchical coarse-to-fine manner. Then, the direction vector for each synthetic view at each pyramid level and for all rotations and scales is computed. Therefore, we first compute the Sobel filter and then use a threshold that removes all edges with a magnitude of less than 0.07 for image intensities of a normalized image. These computations take about 14 seconds on an iPhone4 and need to be done only once offline.

In what follows, we will first describe the procedure carried out to find the transformation of the template in each current camera image without incorporating gravity measurements. We will then show how a gravity-aware approach is able to overcome the limitations of the standard approach. For each current camera image, we first create an image pyramid with downscaled versions of the image and then compute the direction vector for each pyramid level in the same manner as for the reference template.

If we do not have an approximate scale, position and orientation of the template in the current image, e.g. from the last camera image, we start by finding the maximum of the similarity measure in the transformation space in the highest pyramid level using exhaustive search. We use the same similarity measure as in [24]. Given this coarse transformation, we then track the highest similarity down the pyramid by computing the similarity measures with the neighborhood in transformation space for each pyramid level. When tracking multiple frames where the current transformation of the template can be considered similar to the one in the last current image, we skip the exhaustive search on the highest pyramid level and use the transformation from the last frame as a starting point instead.

The explained algorithm requires the precomputed direction vectors of all the possible warpings of the template image to be quickly accessible. The only solution to this on mobile devices is to store them in main memory. Depending on the size of the template image used and the sampling resolution of the transformation space, the direction vectors easily require 200 MB of main memory. As the memory consumption grows exponentially with the supported

degrees of freedom, it is not feasible to support perspective transformations, which would increase the dimensions of the precomputed transformation space from 2 (i.e. scale and in-plane rotation) to 4 (i.e. scale and all 3 rotation axis). Assuming 60 steps per DoF this would result in $60^4 = 12,960,000$ instead of $60^2 = 3,600$ precomputed direction vectors. This will not only increase computational complexity but more critically exceed the memory available on current mobile devices, making 6 DoF tracking impossible with this kind of algorithms.

We propose to enable 6 DoF pose estimation from a horizontal template by using the 4 DoF edge-based detection algorithm and considering the measured gravity vector. Given a current camera image, taken under an arbitrary view angle that is not perpendicular to the template, the edge-based tracker is not able to find the correct transformation, as it supports similarity transforms only, cf. figure 11a. While the illustrated transformation is the one providing the highest similarity in the class of similarity transforms, it is clearly wrong. We propose to first rectify the camera image based on the gravity as described in section 4.2. We then proceed as described above on the gravity-rectified camera image which will result in a similarity transform between the template image and the rectified camera image, see figure 11b. By applying the inverse of the transformation used to rectify the camera image to the four corners found in the rectified image, we gain the position of the corners in the original image, as displayed in figure 11c. These, together with the intrinsic camera parameters, can then be used to compute a full 6 DoF camera pose which can be used to display virtual 3D objects spatially registered with the template as shown in figure 11e. In figure 11d, we see that similarity tracking with no rectification results in incorrect augmentation.

Figure 1 shows this algorithm running on an iPhone 4 in the right image. A colorful logo is correctly tracked, even though it is partially occluded by a ballpen.

6 Inertial Sensor-Enabled Gravity-Aware Augmentations

Above, we presented different ways of aiding computer vision algorithms that are used for camera pose estimation. The benefit of an improved camera pose for handheld AR becomes visible, when augmenting the camera image with virtual 3D objects. The more accurate and stable the estimated pose is, the better and more realistic the augmentation becomes. In this section, we present an approach using the knowledge of the measured direction of the gravity to directly improve augmentations in handheld AR. We show that realism and plausibility of augmented 3D models can be increased by adding simple physics to the model that are based on the measured gravity vector. The gravitational force influences the visual appearance of objects in the real world. Therefore, it makes sense to simulate the effect of the gravity also in the virtual part of an Augmented or Mixed Reality scene.

In classical video-see-through AR applications, there exist two coordinate systems where one is associated to the tracked object and the other one is attached to the camera. The aim of camera pose tracking is to find the transformation that transforms the virtual objects from the object coordinate system into the camera coordinate system in order to render them. If the camera is static, as for instance the case for most kiosk AR applications, the gravity with respect to the camera is constant and can be defined offline. In handheld AR applications that track a static object with a known orientation, the gravity is constant with respect to the coordinate system of the object and can therefore be computed from the camera pose.

However, in handheld AR applications that track a real object, which may freely move and rotate, neither of the above mentioned approaches works. As shown in [19] for an inertial sensor-equipped HMD, the orientation of a visually tracked object with respect to gravity can be computed given the direction of the gravity in the camera coordinate system. While they use the direction to control a game, we propose to use this information in order to improve the visual appearance of the virtual augmentation attached to a freely moving object. The pose used to render gravity-aligned parts in an augmentation combines the translation given from the visual tracking and the rotation from the gravity measurement.

Figure 12: A rigid 3D model of a candle looks plausible while in a vertical orientation (a). When tilted, the fire of the rigid model rotates with the candle resulting in an unrealistic appearance (b). Aligning the fire with the measured gravity gives a believable impression (c).

Figure 12 compares the augmentation of a static 3D object representing a candle on a muffin with a gravity-aware version where the fire is aligned with the direction of the gravity. As long as the plate, which is the real object being tracked in this case, is located on a horizontal plane as shown in figure 12a, the static augmentation appears plausible. However, when it is rotated as in figure 12b, the static augmentation looks unrealistic which completely destroys the illusion of Augmented Reality. Figure 12c, however, provides a realistic appearance thanks to inertial sensor-enabled gravity-aware augmentations. While this example aligns the articulated part contrary to the direction of the gravity, most other example applications, such as a tire swing mounted to a tree, would align the articulated parts downwards aligned with gravity.

We believe, that besides robust and fast tracking, high quality and plausible 3D augmentations are critical for future handheld AR applications.

7 Conclusions and Future Work

In this paper, we presented novel approaches to use the direction of the gravity measured with inertial sensors to improve different parts in the pipeline of handheld AR applications.

While the improvements for feature descriptors on both (close to) vertical and (close to) horizontal surfaces could be clearly shown by the evaluation results, our proposal to perform affine gradient-decent tracking on gravity-rectified images at least outperforms regular homography-based approaches in theory. Unfortunately the sensor hardware of current mobile devices is not accurate enough for this approach, yet. However, we could show, that gravity-rectification of the camera image allows for performing an edge-based detection algorithm which enables 6 DoF camera pose estimation, while running a naïve 6 DoF implementation of this method is impossible on modern handheld devices. Our contribution to increase realism of augmentations is obvious, as shown in figure 12c, and the reader is invited to refer to the supplementary video to get a better impression of the improvement.

All proposed techniques inherently require a handheld device with attached inertial sensors. However, as virtually all modern off-the-shelf mobile phones and tablets are equipped with such sensors, we do not consider this a very limiting constraint. In the near future, inertial sensors will be ubiquitous in any electronic devices. Another assumption for the improved approaches to camera pose

tracking presented in this paper, is that the orientation of the environment to track is known and static with respect to gravity. Looking at actual handheld AR applications developed by us and others, this is quite often the case.

Obviously, the presented methods can handle surfaces that are not perfectly horizontal or vertical up to a certain extent. While we did not evaluate this tolerance on horizontal surfaces in this paper, the performance of GAFD for non-upright surfaces has been studied in [12]. All techniques presented in this paper run in real-time on current mobile devices as can be seen in the supplementary video. Computing GAFD is less costly than regular rotation-invariant feature descriptors as the expensive step of computing the canonical feature orientation, e.g. as the dominant gradient direction, can be skipped here. However, all approaches presented in this paper that are based on gravity-rectified camera images introduce an additional image warping to the algorithm. It depends on the implementation of the warping and the method applied to the gravity-rectified image if the overall method is faster than a method not using this approach. In any case, we have shown in section 5.2 that gravity-rectification enables methods that would not be possible without it.

All approaches presented in this work use the direction of the gravity, which describes a two-dimensional rotation. Many handheld devices are, in addition to inertial sensors, equipped with a digital compass providing measurements of the third rotational dimension. Taking into account this rotation in the proposed methods is quite straight forward. However, we doubt this is applicable for two simple reasons. First, digital compasses are inaccurate and highly dependent on the surrounding, making them very unreliable. The second reason is, that while the orientation with respect to gravity is constant and known for many objects, the orientation with respect to north is not. Cars, billboards and TV screens are usually orientated vertically and upright. Their point on the compass, however, is arbitrary and changes among different instances of the same item. The same applies for horizontal objects on a table or on the floor.

While all algorithms and evaluations in this paper were performed for tracking planar templates, in future, handheld AR will also make use of 3D natural feature tracking. This opens up new challenges and opportunities to support vision-based tracking with additional sensor data. Even more interesting will be consumer handheld devices with a stereo camera that very recently entered the market. In the future, handhelds might even be equipped with active infrared illumination to allow for real-time depth estimation, similar to Microsoft's Kinect. We believe that these types of new sensors will lead to exciting novel computer vision algorithms, which will also be able to benefit from knowing the direction of the gravity provided by inertial sensors.

REFERENCES

[1] S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[2] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal on Computer Vision*, 56(3):221–255, 2004.

[3] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[4] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. *Computer & Graphics*, 33, 2009.

[5] O. Chum and J. Matas. Matching with prosac - progressive sample consensus. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[6] E. Eyjolfsdottir and M. Turk. Multisensory embedded pose estimation. In *Proc. IEEE Workshop on Applications of Computer Vision*, 2011.

[7] N. Hagbi, O. Bergig, J. El-Sana, and M. Billinghurst. Shape recognition and pose estimation for mobile augmented reality. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2009.

[8] S. Hinterstoisser, V. Lepetit, S. Benhimane, P. Fua, and N. Navab. Learning real-time perspective patch rectification. *International Journal of Computer Vision*, 91(1):107–130, 2011.

[9] G. Klein and T. Drummond. Sensor fusion and occlusion refinement for tablet-based AR. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2004.

[10] D. Kotake, K. Satoh, S. Uchiyama, and H. Yamamoto. A hybrid and linear registration method utilizing inclination constraint. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2005.

[11] D. Kotake, K. Satoh, S. Uchiyama, and H. Yamamoto. A fast initialization method for edge-based registration using an inclination constraint. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2007.

[12] D. Kurz and S. Benhimane. Inertial sensor-aligned visual feature descriptors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[13] W. Lee, Y. Park, V. Lepetit, and W. Woo. Point-and-Shoot for Ubiquitous Tagging on Mobile Phones. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2010.

[14] S. Lieberknecht, S. Benhimane, P. Meier, and N. Navab. A dataset and evaluation methodology for template-based tracking algorithms. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2009.

[15] J. Lobo and J. Dias. Vision and inertial sensor cooperation, using gravity as a vertical reference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12), 2003.

[16] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision*, 60(2):91–110, 2004.

[17] B. Lucas and T. Kanade. An iterative image registration technique with application to stereo vision. In *International Joint Conference on Artificial Intelligence*, 1981.

[18] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10), 2005.

[19] O. Oda and S. Feiner. Rolling and shooting: two augmented reality games. In *Proc. ACM International Conference on Human Factors in Computing Systems (Extended Abstracts)*, 2010.

[20] M. Özuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[21] G. Reitmayr and T. Drummond. Going out: robust model-based tracking for outdoor augmented reality. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2006.

[22] K. Satoh, M. Anabuki, H. Yamamoto, and H. Tamura. A hybrid registration method for outdoor augmented reality. In *Proc. International Symposium on Augmented Reality*, 2001.

[23] H. Shum and R. Szeliski. Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision*, 36(2):101–130, 2000.

[24] C. Steger. Occlusion, clutter, and illumination invariant object recognition. *International Archives of Photogrammetry and Remote Sensing*, 34(3), 2002.

[25] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *Proc. IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2008.

[26] D. Wagner and D. Schmalstieg. First steps towards handheld augmented reality. In *Proc. 7th IEEE International Symposium on Wearable Computers*, 2003.

[27] C. Wu, B. Clipp, X. Li, J.-M. Frahm, and M. Pollefeys. 3d model matching with viewpoint-invariant patches (vip). In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.