To appear in the Proceedings of the 9th Computer and Robot Vision Conference (CRV'12), Toronto, Canada, May 2012.

Accelerometer Localization in the View of a Stationary Camera

Sebastian Stein and Stephen J. McKenna
School of Computing
University of Dundee
Dundee, United Kingdom
{sstein|stephen}@computing.dundee.ac.uk

***Abstract*—This paper addresses the problem of localizing an accelerometer in the view of a stationary camera as a first step towards multi-modal activity recognition. This problem is challenging as accelerometers are visually occluded, they measure proper acceleration including effects of gravity and their orientation is unknown and changes over time relative to camera viewpoint. Accelerometers are localized by matching acceleration estimated along visual point trajectories to accelerometer data. Trajectories are constructed from point feature tracking (KLT) and by grid sampling from a dense flow field. We also construct 3D trajectories with visual depth information. The similarity between accelerometer data and a trajectory is computed by counting the number of frames in which the norms of accelerations in both sequences exceed a threshold. For quantitative evaluation we collected a challenging dataset consisting of video and accelerometer data of a person preparing a mixed salad with accelerometer-equipped kitchen utensils. Trajectories from dense optical flow yielded a higher localization accuracy compared to point feature tracking.**

***Keywords*-computer vision; inertial sensors; sensor fusion; accelerometer detection; localization; tracking**

I. Introduction

With accelerometers becoming increasingly ubiquitous (through, e.g., smartphones and tablets) there is a growing interest in fusing accelerometer data with visual data. Combining these sensor types may have strong potential as they provide complementary information. While accelerometers capture subtleties in the movement of the device, computer vision may, for example, put this information into spatial context and into relation with other entities. In this paper we consider the problem of localizing an accelerometer in the visual field of a stationary camera in the context of situational support systems.

In aging societies of many countries the ratio of people needing personal care to those able to provide care steadily increases. Therefore, the role of technology to develop new assistive solutions gains in importance. While prototypes of situational support systems are able to track and guide a cognitively impaired person through some activities of daily living (ADL) using embedded sensors, food preparation activities appear to be particularly challenging. We investigate the localization of accelerometers embedded in kitchen utensils in the visual field of a camera mounted to have a top-down view onto a work surface (see Fig. 1). We propose to fuse visual and accelerometer data by first estimating the locations of accelerometers in the camera's frame of reference. In our setup accelerometers are visually occluded at all times. They are embedded into kitchen utensils to unobtrusively capture utensil movements. While for some utensils accelerometers are embedded in their handles, for others the sensors are attached in locations that are not necessarily close to a person's hand (e.g., on the rim of a bowl). In order to solve the localization problem, acceleration data need to be extracted from camera images and compared to data captured from accelerometers. In this context, visual motion estimation may generally be performed at the object level through object tracking or at the point level through either dense optical flow [5] or by tracking point features [23]. We investigate visual acceleration from point feature trajectories and optical flow because these methods do not make strong assumptions about the appearance of

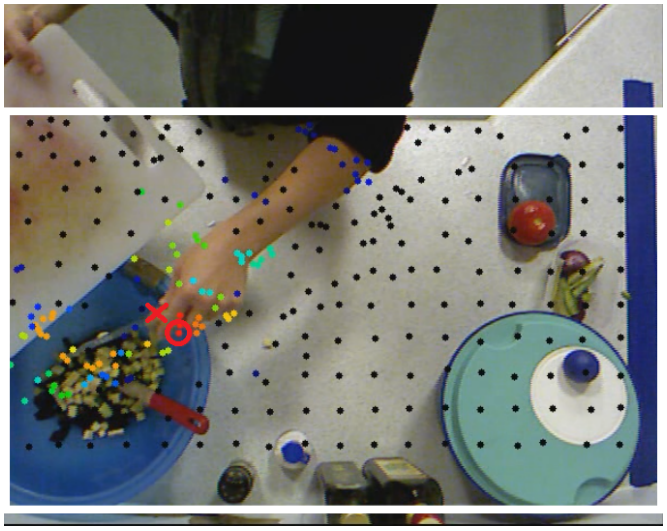

Fig. 1. Accelerometer Localization: Point features (colored dots) are tracked within a region of interest (white rectangle). By measuring similarity of accelerations along these trajectories with accelerometer data (black indicates weakest and red indicates strongest similarity), the algorithm estimates the accelerometer location (red circle). The red cross marks the ground-truth location.

the utensils in which the accelerometers are embedded or about the hands manipulating them. With combined 2D and depth cameras becoming increasingly available at a reasonable price, we use this type of sensor (Kinect) and present preliminary results using trajectories that incorporate depth information.

There are four core problems associated with the task of fusing vision and accelerometer data. (i) An accelerometer measures proper acceleration (relative to free-fall), whereas visual acceleration estimates represent coordinate acceleration relative to a 3D coordinate system attached to the camera. The gravity vector, which is necessary to convert coordinate to proper acceleration, is unknown in the 3D coordinate system attached to the camera. (ii) The orientation of an accelerometer is unknown and changes over time relative to the (stationary) camera viewpoint. With accelerometers that measure only translational and not angular acceleration (such as the devices used in this paper) it is infeasible to calibrate and subsequently track their relative orientation. (iii) Different sensor frequencies of accelerometers and cameras impose another problem as accelerometer frames and images are not acquired at the same time, even if the streams are synchronized. (iv) Although camera frames and accelerometer readings may be timestamped, we may not generally assume that these clocks are synchronized. Similarly, if timestamps are generated by a server receiving new data from the client devices (cameras and accelerometers), the timestamps are synchronized, but they do not reflect latencies in data transmission.

This paper makes the following contributions. It proposes an accelerometer localization pipeline that may be combined with different techniques for constructing point feature trajectories from visual data. Here, we investigate trajectories from sparse optical flow (KLT) and trajectories sampled on a regular grid from a dense flow field [5]. An incremental similarity measure for matching visual trajectories with accelerometer data is used that is easy to implement, fast to compute and outperforms the state-of-the-art. We collected and labeled a realistic and challenging dataset for quantitative evaluation of localization accuracy. This dataset consists of video and accelerometer data of a person preparing a mixed salad with three accelerometer-equipped kitchen utensils and will be made public. We are the first to present quantitative data on accuracy of accelerometer localization in video. We compared KLT and dense optical flow for trajectory construction, and two measures of similarity between accelerometer data and acceleration along point trajectories (i.e. normalized cross-correlation and the measure proposed in this paper).

II. Related Work

Inertial sensors have been used for several pervasive computing applications including self-localization [7], [12], action recognition [17], [18] and skill assessment [9]. Pham et al. [18] recognized actions such as chopping, peeling, stirring and scooping from accelerometers embedded into kitchen utensils using dynamic time warping.

In the computer vision community point feature trajectories have been used for action classification [15]. Wang et al. [25] proposed in this context to construct trajectories from dense optical flow, obtaining superior performance compared with trajectories of salient features [20] tracked with KLT [13], [23]. In this paper we compare the same methods for trajectory construction in the context of accelerometer localization and draw similar conclusions.

First prototypes of situational support systems for cognitively impaired people using sensors such as accelerometers and RFID [10] or cameras [11] are tailored to a single ADL task at a time. Hoey et al. [11] used POMDPs [3] to guide a person through the task of hand-washing with a camera mounted above a sink. In [10] the authors proposed a general specification method for prompting systems which was demonstrated on the task of making a cup of tea [16].

The general problem of fusing visual data with accelerometer data and some potential solutions are discussed in detail by Corke et al. [4]. Inertial sensor localization in a camera's field of view is primarily being addressed in the context of tracking people [14], [21], [22]. For example, Teixeira et al. [22] used magnetometers and accelerometers in mobile phones to identify and localize multiple people tracked from CCTV cameras. Sensor measurements from phones carried by the subjects were associated with tracked body positions in a hidden Markov model for localization and to resolve ambiguities in visual tracks. This approach makes strong assumptions about the appearance of the object (person) that is to be tracked. Shigeta et al. [21] make similar assumptions localizing an accelerometer using normalized cross-correlation (NCC) of accelerometer data with trajectories of a jacket and a hand. Maki et al. [14] proposed point feature trajectories in combination with NCC for accelerometer localization without prior knowledge of object appearance. Unfortunately, those authors did not present quantitative data on localization performance.

III. Localization Architecture

A. Overview

The goal is to estimate the location of accelerometers in the field of view of a camera. The data flow from the raw input data to location estimates is illustrated in Fig. 2.

The visual data, i.e. the color frames and depth maps, are temporally and geometrically aligned such that at each pixel, RGB values and depth are available. In practice, there is a small time delay between color and depth frames. With depth from structured light the depth measurement for some pixels is invalid. This is observable at depth discontinuities, at non-opaque or reflective surfaces, and in areas that lie in the shadow of the projected light pattern.

We extract point feature trajectories by tracking either salient features [20] with pyramidal KLT [1] or points on a regular grid with dense optical flow [5]. Both techniques are described in Sec. III-B.

In order to match extracted video features with accelerometer data, locations in image space need to be transformed into world coordinates, which requires the camera to be calibrated. Additionally, locations need to be differentiated twice to

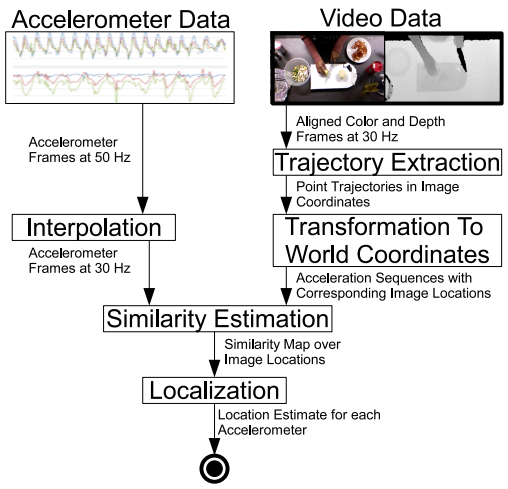

Fig. 2. Localization Architecture: Data flow from raw input streams generated by accelerometers and the camera to location estimates for each accelerometer in the current video frame.

accelerations. In order to transform coordinate acceleration to proper acceleration, the gravity vector needs to be estimated in the view of the camera and added to visual acceleration estimates.

The resulting acceleration sequences are not directly comparable to the captured accelerometer data, as the frequencies of the camera and the accelerometers are different. Therefore, we interpolate the acceleration data from accelerometers to match the intervals between subsequent video frames.

Once these processing steps are completed we are able to estimate the similarity between visual acceleration sequences and the accelerometer data. Here, we compute the similarities exhaustively between all current visual acceleration sequences and all accelerometers. Based on the resulting similarity map between devices and tracklets, we estimate the location of each accelerometer in the current video frame.

B. Feature Extraction

We consider two methods for trajectory construction: sampling the image at salient (highly textured) locations [20], which are tracked over time using sparse optical flow (pyramidal KLT [1]), and sampling the image on a regular grid with point locations updated over time using Farnebäck's dense optical flow algorithm [5]. While the first method extracts trajectories at feature locations that are expected to be easily tracked individually, the latter evenly samples the image space and uses a spatially smooth flow field. Both methods are explained in detail below.

1) Trajectories from feature point tracking: In this first trajectory construction method trajectories are initialized at keypoint locations, defined as image locations whose Hessian matrix has two large eigenvalues [20]. The smaller eigenvalue of the Hessian matrix is called *cornerness*. A fixed maximum number of trajectories N is maintained. From the first frame of a video we initialize a trajectory at the image location with highest cornerness. We proceed by initializing trajectories at locations in order of decreasing cornerness subject to the following constraints. The total number of trajectories is not greater than N, the Euclidean distance of a point to the locations in the current frame of all trajectories is greater than a threshold d, and the cornerness of the point is greater than a fraction of the maximum cornerness value in the current image. The distance threshold d prevents initialization of large numbers of trajectories in a small area of the image.

For each subsequent frame, currently maintained point features are attempted to be tracked using pyramidal KLT [1]. We call the set of trajectories that are tracked successfully to the current frame, or were newly initialized, *active* trajectories. If after an update step there are less than N active trajectories, we initialize new trajectories at keypoint locations (as described above) with these new locations having a minimum Euclidean distance d to all active trajectories.

We used the implementation for feature extraction as proposed by Shi and Tomasi [20], and tracking as proposed by Bouguet [1] provided by the OpenCV library [6].

2) Trajectory construction with grid-sampling and dense optical flow: The second method we employ to construct trajectories from video initializes trajectories at locations on a regular grid with horizontal and vertical spacing d between points. For each new frame we compute a dense flow field as described by Farneback [5] and provided by OpenCV [6]. Point feature locations are then updated through shifting previous locations by the horizontal and vertical flow estimated for their respective positions. In the case where a newly estimated location lies outside the image region the corresponding point trajectory is discarded.

New trajectories are initialized at grid point locations whose minimum distance to current locations of active trajectories is greater than d. To avoid situations in which large number of trajectories *gather* in a small area, we employ a technique to terminate trajectories, which is described in detail in Sec. III-F.

3) Extension to 3D: When transforming image locations into world coordinates we need to provide estimates of distance to the camera for points along trajectories (see Sec. III-C). This distance has an impact on the magnitude of estimated accelerations. In addition to trajectories with a fixed depth, we explore an extension of point feature trajectories into the third dimension through the use of combined 2D and depth cameras. Specifically, we annotate the point locations in trajectories with the measured depth at the point's position. Note that 3D trajectories could also be constructed from scene flow [24]. Estimating dense 3D scene flow, however, is computationally too expensive for online accelerometer localization [8].

In practical scenarios the depth of some pixels in a frame may become invalid. This happens for example frequently with cameras that use structured light for constructing depth maps. If the situation occurs that the depth at the current image location of a point feature trajectory is invalid, we provide a depth estimate from previous depth values and velocities. In the case where a velocity estimate for the trajectory exists, we

estimate the current depth from the previous depth, the estimated velocity and the video frame rate by linear prediction. If no estimated velocity is available we assume the depth to be stationary.

C. Transformation to World Coordinates

In order to match acceleration sequences captured by accelerometers, which are measured in terms of meters per second squared (m/s^2), the image locations of point trajectories, which are represented in terms of pixels, need to be transformed into world coordinates represented in meters by using the camera's intrinsic parameters. Then, through differencing and with known video frame rate, we derive metric velocities and accelerations. As accelerometers measure proper acceleration as opposed to coordinate acceleration measured by a camera, we calibrate the camera for the direction of gravity and superimpose gravitational effects on the acceleration sequences estimated from point feature trajectories.

We used functions provided by OpenCV [6] to determine intrinsic parameters as proposed by Zhang [26] and distortion coefficients as proposed by Brown [2] from several views of a chessboard pattern. The library also provides functions to calculate undistorted image locations that we use when transforming image locations into world coordinates.

Let the focal length be f, the dimensions of the image elements be s_x, s_y and the principal point be (c_x, c_y). Given an undistorted image location (x, y) and an associated depth z (either from a depth map or fixed), we transform locations on trajectories from image into world coordinates as follows:

$$X = \frac{(x - c_x)z}{fs_x} \qquad Y = \frac{(y - c_y)z}{fs_y} \qquad Z = z \quad (1)$$

D. Visual Acceleration Estimation

Following transformation to world coordinates, trajectories T_i are described as sequences of locations $\mathbf{l}_t : (X_t, Y_t, Z_t)$. Let f_{vid} be the constant video frame rate. Considering a consecutive pair of locations $\mathbf{l}_{t-1}$ and $\mathbf{l}_t$, the velocity $\mathbf{v}_t$ at time t is given by

$$\mathbf{v}_t = (\mathbf{l}_t - \mathbf{l}_{t-1})/f_{vid} \quad (2)$$

The acceleration $\mathbf{a}_t$ at time t given $\mathbf{v}_t$ and $\mathbf{v}_{t-1}$ is computed analogously.

In order to avoid instabilities in the estimated velocities and accelerations, which occur frequently when approximating a differential by discrete differences [19], we smooth locations in world coordinates on each trajectory with a Gaussian with zero mean and some small standard deviation. Note that because of this smoothing procedure, which takes some low number of future locations on a trajectory into account to estimate the current smoothed location, the estimates $\mathbf{v}_t$ and $\mathbf{a}_t$ will only be available with some fixed delay.

As a final processing step before point feature trajectories are compared with device accelerations measured by an accelerometer, we need to add the gravity component to all estimated accelerations $\mathbf{a}_i$. While the magnitude of gravity at the earth's surface is well known (we use standard gravity $|\mathbf{g}| = 9.80665 m/s^2$), the direction relative to the cameras coordinate system depends on the positioning of the camera.

We extract the direction of gravity indirectly from the depth map of surfaces that are assumed to be perpendicular to the direction of gravity, for example a work surface or a floor. We mark a set of at least three points on one of these surfaces on a single depth frame. The marked points are transformed into world coordinates as explained above. Then, the direction of gravity is estimated by forming the co-planar vectors $\mathbf{a} = \mathbf{p}_1 - \mathbf{p}_0$ and $\mathbf{b} = \mathbf{p}_2 - \mathbf{p}_0$, computing the surface normal using the cross-product and normalizing the result. This result may be refined by sampling the points $\mathbf{p}_0$, $\mathbf{p}_1$ and $\mathbf{p}_2$ multiple times from a larger number of marked points and averaging the estimated surface normals or by fitting a plane through these points using e.g. least squares prior to surface normal estimation. The gravity vector $\mathbf{g}$ points in the estimated direction.

The final acceleration estimation $\mathbf{a}'_t$ of a point feature trajectory at time t is given by $\mathbf{a}'_t = \mathbf{a}_t + \mathbf{g}$.

E. Accelerometer Data Interpolation

In the general case the frame rate of a video camera f_{vid} and the rate f_{acc} with which an accelerometer captures its data are not equal. For two acceleration estimates to be comparable, however, they need to correspond to the same distinct point in time. As in our case f_{vid} is lower than f_{acc}, we linearly interpolate acceleration samples captured by the accelerometers. Note that for this procedure to be valid we need to assume that accelerometer data and video data are synchronized.

Once we interpolate the accelerometer frames for all video frames, the resulting streams $A^{acc'} : (\mathbf{a}_0^{acc'}, \dots \mathbf{a}_t^{acc'})$ and $A^{vid} : (\mathbf{a}_{t-j}^{vid}, \dots, \mathbf{a}_t^{vid})$ are comparable and ready to be matched for localization.

F. Similarity Measure

Localizing an accelerometer in the camera's field of view based on acceleration sequences estimated from point feature trajectories involves measuring the similarity between the sequences A^{vid} and $A^{acc'}$ for each trajectory and accelerometer.

We update similarity estimates in an online manner with each video frame. Therefore, we define the initial similarity of an empty sequence with the accelerometer stream to be zero

$$S(A_i^{vid}, A_j^{acc'}) = 0 \quad (3)$$

for each point feature trajectory $i = 0, \dots, N$ and accelerometer $j = 0, \dots, M$. As we do not assume a fixed orientation of the accelerometer in the scene or estimate its pose from data, we consider only the norms of the acceleration vectors. The norms are thresholded to suppress errors caused by sensor measurements, inaccurate visual tracking and imprecise synchronization. If both the acceleration measured by the accelerometer and the acceleration estimated from a point feature trajectory exceed this threshold T^{acc} in the current

frame, the similarity between the corresponding trajectory and the accelerometer is incremented by one. In order to reduce the impact of frames in the past on the similarity, we devise a recursive function with multiplicative temporal decay $\alpha \in [0, 1)$. The resulting similarity measure may be expressed more formally as follows:

$$S_t(A_i^{vid}, A_j^{acc'}) = \alpha \cdot S_{t-1}(A_i^{vid}, A_j^{acc'}) + 1\left[|\mathbf{a}|_t^{vid} \geq T^{acc} \wedge |\mathbf{a}|_t^{acc} \geq T^{acc}\right], \quad (4)$$

where t is the index of the last available video frame and $1[]$ is the indicator function. The similarity between a point feature trajectory and an accelerometer stream is thus defined as the number of frames in which both capture significant acceleration, giving higher weight to the contribution of recent frames.

Recall that trajectories are initialized if the number of active trajectories is too low or if some areas of the camera's field of view are currently not covered by any active trajectory (as explained in Sec. III-B). In the latter case the number of active trajectories may grow indefinitely, which is undesirable. Therefore, we employ a distance threshold $\tau^{terminate}$. If the distance between the locations in the most recent frame of a pair of trajectories is smaller than this threshold, the trajectory that has been tracked for a smaller number of frames is eliminated.

Consider a trajectory that is being newly initialized at a point in time at which other trajectories have already been tracked for a large number of frames. As can be seen from Eq. (4), these trajectories accumulate a potentially high similarity measure over time. To increase the effectiveness of our algorithm in this case, we do not initialize the similarity measure of the new trajectory to zero. Instead, we set its similarity equal to the similarity of the trajectory corresponding to the location of the closest tracked point.

G. Localization

Finally, we estimate the location of an accelerometer as the location in the most recent frame of the point feature trajectory with highest similarity. In rare cases in which more than one active trajectory has the same similarity (usually occurring at initialization), the algorithm does not give a location estimate. Despite its simplicity, the localization algorithm performs reasonably well in practical scenarios (see Sec. IV-C).

IV. Evaluation

In this section we present evaluation results for the localization algorithm described in Sec. III above. We start by explaining the experimental setup and describing how the dataset was collected, synchronized and annotated.

A. Experimental Setup

We collected data of a person preparing a salad in our laboratory kitchenette (see Fig. 3). We used wireless accelerometers developed in CultureLab at Newcastle University, which capture translational acceleration along 3-axes at 50Hz

Fig. 3. Experimental Setup: Laboratory kitchenette with Kinect facing the work surface and accelerometers attached to utensils with tape (top left).

with 16-bits per axis. The data are transmitted over IEEE 802.15.4-2006 radio. All frames are timestamped upon arrival at the server. The recorded times are therefore subject to jitter and usually do not exactly correspond to the times at which the acceleration measurements were actually captured.

As visual sensor we used a Microsoft Kinect which has an RGB camera and produces depth maps from near-infrared structured light. The depth maps have a per pixel resolution of 11 bits. A non-linear function maps these depth values to metric units. An operating range of e.g. $[1m, 3m]$ is therefore represented by only about 250 distinct values. Color and depth images have a spatial resolution of 640x480 pixels and are produced at about 30 frames per second. We recorded timestamped color and depth images.

The Kinect was mounted to have a top-down view of the work surface. It was attached to an adjustable arm on a vertical bar, providing a high degree of flexibility in positioning. It was connected to a laptop computer on which the recording software was executed. We adjusted the camera manually to roughly align the work surface with the image plane. At an operating height (distance between sensor and work surface) of 100cm and with 22cm distance to the wall, the camera captured a 108cm wide area of the work surface.

B. Dataset

The dataset contains 13,263 frames of combined color and depth data and 31,346 frames of 3-axis acceleration data capturing a person preparing a mixed salad. Note that the acceleration data were captured by a total of three devices some of which were sometimes moved at the same time (e.g., the spoon and the bowl move concurrently while the salad is being mixed before serving). The utensils with attached accelerometers were a spoon, a plastic bowl and a knife. A chopping board was also used in this experiment but was not equipped with an accelerometer. Ten ingredients were used in total to prepare the salad: balsamic vinegar, beetroot, cheese, courgette, green salad, olive oil, pepper, red onion, salt and

ID	Description	#Frames
1	Knife Synchronization Signal	203
2	Moving Spoon From Bowl To Worktop	25
3	Spoon Mixing Dressing	310
4	Knife Slicing Beetroot	296
5	Knife Scraping Beetroot Into Bowl	125
6	Knife Scraping Courgette Into Bowl	45
7	Knife Scraping Cheese Into Bowl	106
8	Knife Scraping Green Salad Into Bowl	85
9	Knife Slicing Green Salad	45
10	Knife Slicing Green Salad 2	170
11	Knife Scraping Green Salad Into Bowl	123
12	Knife Dicing Tomatoes And Moving Into Bowl	1133
13	Spoon Mixing Salad	375
14	Spoon Mixing Salad 2	95
15	Spoon Serving Salad Into Plates	798
16	Knife Synchronization Signal 2	233
	Total	4167

TABLE I
TEST SEQUENCES: SUBSEQUENCES WITH AT LEAST ONE ACCELEROMETER MEASURING SUBSTANTIAL ACCELERATIONS OVER AN EXTENDED PERIOD OF TIME.

tomato. Ingredients did not carry any sensors or tags.

As the timestamps of video and accelerometer frames were not synchronized, we synchronized the data as a preprocessing step by establishing correspondences in both data streams. For this purpose the experimenter hit the knife with an accelerometer attached to it repeatedly on the work surface producing strong signals in both the accelerometer and the video (depth) stream; five times before and after the experiment. An annotator manually marked the correspondences at the start and the end of the experiment separately. Using least squares estimation, we estimated one time-offset for the start and one offset for the end of the sequence. As these offsets tend to diverge, we linearly interpolated timestamp correspondences between the start and end points of the sequence. Note that synchronization could be more easily accomplished with cameras and accelerometers that can be triggered. However, we did not use such specialized hardware.

In order to evaluate the performance of our accelerometer localization algorithm quantitatively, we annotated the locations of the three accelerometers in every frame in the dataset. For each pair of video frame and accelerometer, the annotator clicked on the image at the estimated geometric center of the device. As accelerometers were covered by tape and frequently occluded by the participant's hand, we cannot assume these labels to be exact. Nevertheless, from visual inspection we can confirm that the marked locations were reasonably close to the true center locations for our purposes.

For testing our localization algorithm we identified 16 subsequences in the video in which at least one of the accelerometers measured substantial accelerations over an extended period of time. These subsequences are listed in Table I. The localization performances reported in the following sections were based on testing each of these subsequences separately, re-initializing the algorithm at the start of each subsequence.

C. Experiments

For quantitative evaluation of different algorithm configurations we use the average per frame Euclidean distance of an estimated accelerometer location to the ground truth point in terms of pixels in the image plane.

1) Sparse vs. Dense Optical Flow: For the first stage of our localization pipeline (see Fig. 2) we compare two different methods to extract point feature trajectories from video data: sparse point feature tracking and dense optical flow (see Sec. III-B).

For KLT tracking we set the maximum number of active trajectories to $N = 96$ and the minimum distance between trajectories to $d = 14$ pixels. For trajectories from dense optical flow we set the distance between grid points to $d = 24$ pixels, initializing 336 trajectories at equidistant locations. These values have shown good performance in preliminary evaluation experiments not reported here.

Intuitively, a larger number of trajectories is expected to increase localization accuracy as *the true trajectory* is more likely to be found in the sample. Nevertheless, in preliminary experiments, using more than 96 active trajectories with KLT and more than 336 initial trajectories with dense optical flow has not been beneficial to localization performance. Presumably a lower number of trajectories increases robustness against artifacts in the estimated flow fields.

Locations were smoothed over time for estimating visual accelerations with a Gaussian with standard deviation $\sigma = \frac{0.3}{f_{vid}}$. For measuring similarity, we used a temporal decay of $\alpha = 0.9982$ and terminated trajectories from dense optical flow with a threshold distance of $\tau^{terminate} = 5$ pixels (see Sec. III-F). Table II shows evaluation results for these configurations using the depth provided by the camera and using a fixed depth, $\hat{z} = 0.9m$. The depth was chosen based on a distance of the work surface to the camera of about $1.1m$. Recall that this distance is used to convert image into world (metric) coordinates in Eq. (1). While the region of interest as indicated in Fig. 1 is used to compare effects of distance estimation on localization performance, we also report results based on the entire image region (ROI off) with fixed depth.

There are three major observations resulting from these experiments. (i) Trajectories constructed from dense optical flow estimates are shown to perform significantly better than those from sparse tracking. We attribute this substantial difference in localization performance to the smoothness of the dense flow field as opposed to frequently occurring false point-feature correspondences with KLT. (ii) The 3D extension of trajectories as proposed here - annotating trajectories with the depth from the Kinect - does not outperform localization based on a user-defined constant depth. Acceleration sequences constructed with measured depth performed significantly worse than those assuming a constant depth, for trajectories from KLT tracking and dense optical flow alike. This might be due to the fact that the depth maps produced by the camera are not reliable enough for our purposes. Holes in the depth maps on areas that lie in the shadow of the structured light pattern, and point trajectories frequently crossing depth-discontinuities,

ID	#Frames	KLT Depth fixed	KLT Depth var.	KLT ROI off	Dense Optical Flow Depth fixed	Dense Optical Flow Depth var.	Dense Optical Flow ROI off
1	203	74	56	65	32	50	42
2	25	206	313	217	92	99	182
3	310	23	28	24	25	35	33
4	296	19	28	59	49	32	61
5	125	143	215	139	53	46	66
6	45	89	259	118	67	94	152
7	106	67	79	128	89	84	67
8	85	138	146	181	48	69	73
9	45	131	213	71	92	74	165
10	170	66	304	210	36	91	196
11	123	65	81	132	93	114	96
12	1133	60	184	209	23	179	20
13	375	62	63	66	97	105	173
14	95	81	87	235	54	76	150
15	798	135	265	141	52	104	108
16	233	38	338	57	74	85	80
Total	4167	76	167	134	49	106	79

TABLE II
SPARSE VS. DENSE OPTICAL FLOW: DIFFERENT TRAJECTORY CONSTRUCTION METHODS WITH DEPTH FROM THE CAMERA (VARIABLE) AND HARD-CODED (FIXED). PERFORMANCE IS REPORTED AS AVERAGE EUCLIDEAN DISTANCE OF ESTIMATED ACCELEROMETER LOCATION TO GROUND TRUTH IN PIXELS.

might be major issues in this context. A more sophisticated method of extending point feature trajectories to the 3D case, in which depth-discontinuities and noise in the raw depth maps are specifically taken into account, could improve these results, especially for scenarios in which the accelerometer is moved along the camera's view axis. (iii) Narrowing down the location search space with a smaller region of interest in the image increases localization accuracy as expected. While the whole image region also covers some space in front of the work surface, which is in large parts occupied by the participant's body, this region is completely cropped in the latter configuration, eliminating a large fraction of sources of confusion for the localization algorithm. Based on these results we continue our evaluation solely with trajectories constructed from dense optical flow with fixed depth applied to the ROI.

2) Temporal Decay: While it is obvious that we need to limit the effect of frames far in the past on the present similarity measure, there are several different ways to model this behavior. We propose a temporal decay that reduces the impact of such frames gradually. Fig. 4 shows localization accuracy as a function of various values for the decay parameter α. As can be seen, localization accuracy decreases drastically when the decay is not chosen to be close to 1. The value of the parameter $\alpha = 0.9982$ used in the previous section showed the best performance in our experiments. The performance gain compared to no temporal decay was rather small (1.04 pixels on average), which might be due to the limited length of the test-sequences. We expect to see more significant improvement when running the accelerometer localization algorithm over an extended period of time.

3) Comparison to Normalized Cross-Correlation: We also compared our similarity measure to normalized cross-correlation (NCC) as used by Maki et al. [14]. For a fair comparison between NCC and the proposed similarity measure we applied NCC to trajectories constructed from dense optical flow. Estimated accelerations along point trajectories were correlated with the interpolated acceleration data captured by an accelerometer. Results of accelerometer localization using NCC with varying length of temporal sliding window are presented in Table III. Localization accuracy reached its maximum at a window size of about 150 frames with an average Euclidean distance to the ground-truth of 114 pixels. This time window corresponds to an interval of five seconds.

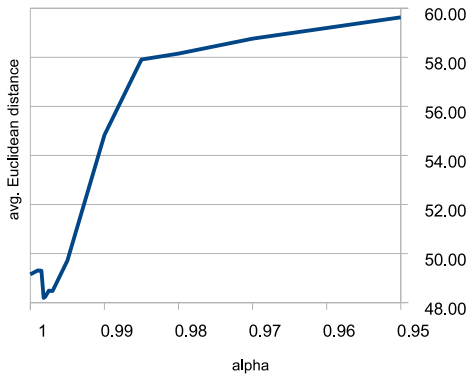

Fig. 4. Temporal Decay: The contribution of past frames to the similarity measure is reduced by a multiplicative factor α. The average Euclidean distance of estimated accelerometer location to the ground-truth in pixels here is the total score over all subsequences. The result of each test is weighted by the number of frames in the corresponding subsequence.

V. CONCLUSION & FUTURE WORK

In this paper we presented a new method for localizing an accelerometer in the field of view of a stationary camera, which is easy to implement and fast to compute. We quantitatively evaluated our algorithm confirming superior performance of trajectories from dense optical flow in the context of accelerometer localization. The proposed method shows higher localization accuracy than the state-of-the-art. Given the significant difference of the best performance achieved with the proposed similarity measure compared to NCC (49 vs. 114 pixels) we conclude that the proposed method is better suited to localizing an accelerometer in a camera's view. We consider a distance between the estimated accelerometer location from the ground truth of about 49 pixels on average, which corresponds to about 8cm at a distance of 90cm from the camera, to be a good localization result, especially because this average value also contains the distances of outliers.

Additionally, we presented a first attempt at extending point feature trajectories to 3D. Our evaluation suggests that a more effective way of combining color and depth information is necessary in this regard. Arguably, the performance measure used here provides little evidence to the frequency with which an estimated accelerometer location actually lies on the device that is to be localized or on an object that is physically connected to the accelerometer. In future work we plan to

	NCC Sliding Window Size											
	10	15	20	30	40	50	60	80	100	150	200	300
Total	199.28	188.10	184.25	163.78	150.06	138.81	131.07	121.79	117.01	114.46	114.93	115.00

TABLE III
NORMALIZED CROSS-CORRELATION: LOCALIZATION ACCURACY WITH VARYING LENGTH OF THE TEMPORAL SLIDING WINDOW. THE AVERAGE EUCLIDEAN DISTANCE OF ESTIMATED ACCELEROMETER LOCATION TO THE GROUND-TRUTH IN PIXELS HERE IS THE AVERAGE OVER ALL FRAMES IN ALL SUBSEQUENCES.

use more informative measures.

By enforcing a unique point estimate for the location of an accelerometer we discard large parts of useful information provided by a similarity map. If for example none of the point trajectories show a strong similarity with an accelerometer after an extended period of time, this could indicate that the accelerometer is outside the camera's field of view. If, on the other hand, trajectories in two (or more) distinct image locations show strong and almost equal similarity to an accelerometer, both location hypotheses should be evaluated. If this localization algorithm is used as a low-level component of, e.g., an activity recognition system it would be useful to utilize the estimated similarity maps as inputs to higher-level processing modules.

In future work we will extend our approach to localizing multiple accelerometers and investigate long-term tracking behavior. In the longer term we plan to automatically recognize interactions of kitchen utensils and ingredients using the locations of accelerometers in the view of a camera.

VI. ACKNOWLEDGEMENTS

This research is funded by an EPSRC studentship from the SiDE hub in Newcastle, from December 2010 until May 2014. We would like to thank Patrick Olivier and his group for providing accelerometers and for useful discussions. We also thank Jesse Hoey for his helpful advice.

REFERENCES

[1] Jean-Yves Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. In *Proceedings of the USENIX Annual Technical Conference, Monterey, California, USA*, 1999.

[2] D. C. Brown. Close-range camera calibration. *Photometric Engineering*, 37(8):855–866, 1971.

[3] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the twelfth Conference on Artificial Intelligence (AAAI'94), Seattle, Washington, USA*, 1994.

[4] P. Corke, J. Lobo, and J. Dias. An introduction to inertial and visual sensing. *International Journal of Robotics Research*, 26(6):519–535, 2007.

[5] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, LNCS 2749, pages 363–370, Gothenburg, Sweden, June-July 2003.

[6] Willow Garage. *OpenCV*, 2012 (accessed January 06, 2012). http://opencv.willowgarage.com/wiki/.

[7] P. Goel, S. I. Roumeliotis, and G. S. Sukhatme. Robust localization using relative and absolute position estimates. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'99), Kyongju, South Korea*, pages 1134–1140, 1999.

[8] S. Hadfield and R. Bowden. Kinecting the dots: Particle based scene flow from depth sensors. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'11), Barcelona, Spain*, 2011.

[9] N. Hammerla, T. Plotz, P. Andras, and P. Olivier. Assessing motor performance with PCA. In *Proceedings of the International Workshop on Frontiers in Activity Recognition using Pervasive Sensing*, pages 18–23, 2011.

[10] J. Hoey, T. Ploetz, D. Jackson, A. Monk, C. Pham, and P. Oliver. Rapid specification and automated generation of prompting systems to assist people with dementia. *Pervasive and Mobile Computing*, 7(3):299–318, 2010.

[11] J. Hoey, A. v. Bertoldi, P. Poupart, and A. Mihailidis. Assisting persons with dementia during handwashing using a partially observable markov decision process. In *Proceedings of the International Conference on Computer Vision Systems (ICVS 2010), Bielefeld, Germany*, 2007.

[12] Ching-Hsien Hsu and Chia-Hao Yu. An accelerometer based approach for indoor localization. In *Proceedings of the 2009 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing (UIC-ATC'09), Brisbane, Queensland, Australia*, pages 223–227, Washington, DC, USA, 2009.

[13] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'81), Vancouver, British Columbia, Canada*, 1981.

[14] Y. Maki, S. Kagami, and K. Hashimoto. Accelerometer detection in a camera view based on feature point tracking. In *Proceedings of the IEEE/SICE International Symposium on System Integration, Sendai, Japan*, 2010.

[15] R. Messing, C. Pal, and H. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *Proceedings of the International Conference on Computer Vision (ICCV'09), Kyoto, Japan*, 2009.

[16] P. Oliver, A. M. Monk, J. Hoey, and G. Xu. Ambient kitchen: Designing situated services using a high fidelity prototyping environment. *Workshop on Affect & Behaviour Related Assistance in the Support of the Elderly (PETRA-09), Corfu, Greece*, 2009.

[17] C. Pham and P. Oliver. Slice&Dice: recognizing food preparation activities using embedded accelerometers. *Ambient Intelligence, LNCS*, 5859:34–43, 2009.

[18] C. Pham, T. Plötz, and P. Oliver. A dynamic time warping approach to real-time activity recognition for food preparation. *Ambient Intelligence, LNCS*, 6439:21–30, 2010.

[19] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *International Journal of Computer Vision*, 50(2):203 – 226, 2002.

[20] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94), Seattle, Washington, USA*, 1994.

[21] O. Shigeta, S. Kagami, and K. Hashimoto. Identifying a moving object with an accelerometer in a camera view. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Nice, France*, 2008.

[22] Thiago Teixeira, Deokwoo Jung, and Andreas Savvides. Tasking networked CCTV cameras and mobile phones to identify and localize multiple people. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, Ubicomp '10, 2010.

[23] C. Tomasi and T. Kanade. Detection and tracking of point features. *Carnegie Mellon University Technical Report CMU-CS-91-132, April*, 1991.

[24] S. Vedula, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(3):475–480, 2005.

[25] H. Wang, A. Klaser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Proceedings of the IEEE International Conference*

on Computer Vision and Pattern Recognition (CVPR'11), Colorado Springs, Colorado, USA, 2011.

[26] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.