Determining Relative Spatial Alignment based on Shared Acceleration Time Series

Rene Mayrhofer
JRC u'smile, University of Applied Sciences Upper Austria
Softwarepark 11, 4232 Hagenberg, Austria
rene.mayrhofer@fh-hagenberg.at

Helmut Hlavacs
University of Vienna, Faculty of Computer Science
Währinger Straße 29, 1090 Wien, Austria
helmut.hlavacs@univie.ac.at

Rainhard Dieter Findling
JRC u'smile, University of Applied Sciences Upper Austria
Softwarepark 11, 4232 Hagenberg, Austria
rainhard.findling@fh-hagenberg.at

ABSTRACT

Detecting if two or multiple devices are moved together is an interesting problem for different applications. However, these devices may be aligned arbitrarily with regards to each other, and the three dimensions sampled by their respective local accelerometers can therefore not be directly compared. The typical approach is to ignore all angular components and only compare overall acceleration magnitudes — with the obvious disadvantage of discarding potentially useful information. In this paper, we contribute a method to analytically determine relative spatial alignment of two devices based on their acceleration time series. Our method uses quaternions to compute the optimal rotation with regards to minimizing the mean squared error. The implication is that the reference system of one device can be (locally and independently) aligned with the other, and thus that all three dimensions can consequently be compared for more accurate classification. Based on real-world experimental data from smart phones and smart watches shaken together, we demonstrate the effectiveness of our method and show an improvement in terms of various signal distance metrics when comparing the three axes before and after derotation.

Categories and Subject Descriptors

H.5.m. [**Information Interfaces and Presentation (e.g. HCI)**]: Miscellaneous

Keywords

Accelerometer time series; spatial alignment; quaternion rotation

1. INTRODUCTION

Common movement can be detected from sufficiently similar acceleration sensor data and has interesting applications in mobile and ubiquitous computing. This includes determining if devices are carried by the same user [13] or transported on the same vehicle [15] as well as an interaction method for securely pairing handheld devices[17, 16, 2, 10]. However, such common movement is inherently three-dimensional. In the general case, the relative alignment of two (or multiple) accelerometers embedded in different devices is unknown: similar devices may be rotated arbitrarily with regards to each other and different devices may embed their accelerometers with arbitrary orientations. Therefore, the three dimensions sampled independently will typically not be aligned and are therefore not directly comparable.

A standard approach to deal with this issue is to discard all angular (i.e. directional) information from the 3D vectors and only use their magnitude (i.e. the length of each vector computed in an Euclidean space). This reduces three dimensions to a single one that is invariant concerning orientation. Even when two co-located accelerometers are oriented differently, they will experience similar overall acceleration magnitudes. However, this simple approach discards potentially valuable information that could be helpful in determining if accelerations are sufficiently similar to each other or not (cf. Section 3).

In this paper, we contribute a method to explicitly determine the relative alignment of two mobile devices with regards to each other based on their recorded acceleration time series. The underlying assumption is that both devices are moved (relatively closely) together and therefore share sensor readings that are only offset by 3D rotation but otherwise similar. Specifically, we assume that both devices experienced similar translation and rotation with regards to a common reference system. Our approach uses quaternions (Section 4) to analytically compute optimal rotation between both device reference systems (Section 5) and, based on real-world sensor data, works even in the presence of small distances between the devices and typical sensor noise (Section 6).

2. RELATED WORK

Known applications of common movement presented in the context of ubiquitous computing (e.g. [13, 15, 16, 17, 2, 10, 8]) have so far taken the simple approach and discarded angular information. However, we suggest that all of these could benefit to various degrees from taking this information into account. Especially when used for securing

MoMM2014, 8-10 December, 2014, Kaohsiung, Taiwan.

device communication [16, 17, 2, 8] would this be valuable — any information that is shared between the legitimate devices but not directly available to a potential attacker increases the latter's entropy of the resultant cryptographic key and consequently improves the security level. In [12], position-invariant heuristics for dealing with sensor displacement were suggested to improve movement recognition with a single sensor (accelerometer and/or gyroscope). Our approach complements this work when multiple sensors are in use, e.g. to detect if a mobile phone and a wrist watch describe the same movement and are therefore on the same hand.

Quaternions have been used to minimize the root-mean-squared deviation (RMSD) between solid bodies [5]. We build upon this work by translating it from body rotation to determining the relative alignment of 3D acceleration time series. Another related use of quaternions has been presented for representing orientations in hand and head movement [3].

3. PROBLEM OVERVIEW

Determining if two (or multiple) devices are moved together based on their respective local accelerations can be seen as a classification problem. When they are moved together, sensor noise and systematic error will still lead to (slightly) different sensor time series. When they are moved separately (but for example with similar frequency and amplitude components), they might still be "close" for some similarity measure (cf. [16] for experimental "positive" and "negative" data). The systematic error is intrinsic: even if the devices are held perfectly together and do not move with regards to each other *and* the sensors are perfect and do not exhibit any sampling noise at all, there will still be differences in acceleration time series whenever rotation is part of the shared movement. This is because of different centers, i.e., the physical placement of the respective accelerometers. Think of one accelerometer on the outer curve and the other on the inner curve of a common rotation; they move together without relative movement, but take different paths in 3D that consequently lead to different *local* accelerations.

This issue is independent of the chosen similarity measure and also occurs when only using the magnitude. In fact, discarding angular information makes it even harder to determine that the devices were moved together because locally measured rotational components would in this case be similar, while the magnitudes differ. Recently suggested heuristics explicitly discard accelerometer time series in periods of large rotational movement [12]. We expect classification accuracy to improve noticeably when comparing three dimensions instead of one.

Our approach to retain this 3D information is for devices to – locally and independently of each other – align the two coordinate systems for the subsequent comparison. We analytically determine the "optimal" rotation between these coordinate systems given only the two 3D time series (which are for example exchanged securely using an interlock protocol and session keys [17, 16]) and the assumption of shared movement. In the scope of this paper, we define optimal to minimize the mean squared error between all of the sample points. One method to analytically determine the relative orientation is to use quaternions.

4. QUATERNIONS AND ROTATIONS

It has been know for a long time that quaternions can be used to represent rotations in a three dimensional space. Furthermore, they possess favorable properties like avoiding so-called "gimbal locks" or enabling easy interpolation, something that other approaches like Euler angles and matrix-based rotation do not exhibit. It is thus straight forward to use quaternions to find the optimal rotation between two sets of vectors.

In the following, only the most important aspects of quaternions are presented, and we follow the notation of [5]. More details about quaternions can be found, e.g. in [11]. A quaternion is a tuple $q = (q_0, \mathbf{q})$, with $\mathbf{q} = (q_1, q_2, q_3)'$. Note that like in Matlab/Octave the operator " $'$ " denotes transposition, and all vectors without it are column vectors. Quaternions are essentially a generalization of complex numbers, i.e. a quaternion consists of a real part (q_0) and three imaginary parts (q_1, q_2, q_3). In the area of three dimensional spaces, this imaginary part may take over the part of a 3D-vector. Since quaternions form up an algebraic structure called *ring*, they allow the algebraic operations addition and multiplication, which for $a = (a_0, \mathbf{a})$ and $b = (b_0, \mathbf{b})$ are defined as follows:

$$\begin{aligned} a + b &= (a_0 + b_0, \mathbf{a} + \mathbf{b}) \\ ab &= (a_0 b_0 - \mathbf{a} \cdot \mathbf{a}, a_0 \mathbf{b} + b_0 \mathbf{a} + \mathbf{a} \times \mathbf{b}). \end{aligned} \tag{1}$$

Here $\cdot$ and $\times$ denote the standard dot and cross products known from Euclidean vector spaces. Interestingly, multiplication is associative, but not commutative, i.e. in general $ab \neq ba$. The Matlab/Octave function shown in Listing 1 accepts two quaternions p and q, both represented by 4D-vectors, and computes pq.

Listing 1: Computing pq for two quaternions p and q. Note that the operation is not commutative.

```
function pq = qmul( p, q )
  a0 = p(1); a = p(2:4);
  b0 = q(1); b = q(2:4);
  acb = a0*b + b0*a + cross(a,b);
  pq = [a0*b0-dot(a,b); a0*b + b0*a + cross
      (a,b)];
end
```

Like for complex numbers, a quaternion $q = (q_0, \mathbf{q})$ does have a *conjugate* quaternion q^c which is defined by $q^c = (q_0, -\mathbf{q})$. The conjugate now enables the computation of the norm $|q|$ of a quaternion, which is defined by $|q|^2 = qq^c$. Note that quaternions u with length $|u|^2 = uu^c = 1$ are called *unit* quaternions.

An important subclass of quaternions is given by *pure* quaternions $q = (0, \mathbf{q})$, which are defined to have a zero real part. For pure quaternions, the operations (1) are simplified accordingly.

By using the rules of (1), quaternions now can be used for computing rotations in a 3-dimensional Euclidean space. Each vector $\mathbf{r} = (r_1, r_2, r_3)'$ of the space is represented by a pure quaternion $r = (0, \mathbf{r})$. Rotations in the space then can be characterized by a unit quaternion u by computing

$$\hat{r} = uru^c. \tag{2}$$

Note that $\hat{r} = (0, \hat{\mathbf{r}})$ is again a pure quaternion, whose vector part $\hat{\mathbf{r}}$ equals $\mathbf{r}$ rotated by some angle ϕ and using the rotation axis $\mathbf{u}$, i.e., the vector part of u. The Matlab/Octave function shown in Listing 2 rotates a vector $p = (0, \mathbf{p})$ by the quaternion u.

Listing 2: Rotating vector p ($p = (0, \mathbf{p})'$) by a rotation represented by a unit quaternion u. The result is again a pure quaternion holding the rotated vector in its vector part.

```
function upuc = rotquat( p, u )
  uc = [u(1); -u(2:4)];
  up = qmul(u,p);
  upuc = qmul(up,uc);
end
```

Given a desired rotation axis $\mathbf{a} = (a_1, a_2, a_3)'$ and a rotation angle ϕ, the quaternion u representing this rotation is constructed by

$$u = (cos\frac{\phi}{2}, sin\frac{\phi}{2}\frac{\mathbf{a}}{|a|}). \tag{3}$$

Thus, given this quaternion, using (2) rotates any desired vector $\mathbf{r}$ by angle ϕ and axis $\mathbf{a}$. The Matlab/Octave function shown in Listing 3 constructs a rotation quaternion u from a rotation angle ϕ and a 3D-vector describing the rotation axis.

Listing 3: Computing a rotation quaternion u.

```
function u = quat( phi, axis )
  u = [cos(phi/2); sin(phi/2)*normc([axis
       (1); axis(2); axis(3)])];
end
```

5. THE OPTIMAL ROTATION

In linear algebra, rotations are represented by orthonormal square matrices U, rotating a vector $\mathbf{x}$ is then achieved by multiplying it from the right: $\hat{\mathbf{x}} = U\mathbf{x}$. Given two sets of vectors $\{\mathbf{x}_k\}$ and $\{\mathbf{y}_k\}$ in [5] it has been shown how to compute an optimal rotation U (through the use of quaternions) such that the overall error

$$E := \frac{1}{N}\sum_{k=1}^{N} |U\mathbf{x}_k - \mathbf{y}_k|^2 \tag{4}$$

is minimized. The respective Matlab/Octave function is shown in Listing 4.

Centering both sets by computing the mean vectors $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ of each set, and then subtracting $\bar{\mathbf{x}}$ from each vector of $\{\mathbf{x}_k\}$ and $\bar{\mathbf{y}}$ from each vector of $\{\mathbf{y}_k\}$ will give lower errors but is not necessary for the method to work. Specifically for acceleration time series, subtracting the mean (in practice a moving average computed over sliding time windows) removes the static offset caused by gravity and has previously also been found advantageous for comparing magnitudes (cf. [17, 16]). Like in [5] it is furthermore assumed that X and Y are $3 \times N$ matrices, whose columns hold the vectors $\mathbf{x}_k$ and $\mathbf{y}_k$.

Listing 4: Computing the optimal rotation u given vectors $\mathbf{x}_k$ and $\mathbf{y}_k$.

```
function [E,P] = residuum( X, Y )
  %correlation matrix
  R=X*(Y');
  %matrix holding all rotations
  F=[R(1,1)+R(2,2)+R(3,3), R(2,3)-R(3,2), R
      (3,1)-R(1,3), R(1,2)-R(2,1);
       R(2,3)-R(3,2), R(1,1)-R(2,2)-R(3,3),
           R(1,2)+R(2,1), R(1,3)+R(3,1);
       R(3,1)-R(1,3), R(1,2)+R(2,1), -R
           (1,1)+R(2,2)-R(3,3), R(2,3)+R
           (3,2);
       R(1,2)-R(2,1), R(1,3)+R(3,1), R(2,3)
           +R(3,2), -R(1,1)-R(2,2)+R(3,3)
           ];
  %compute eigenvector evvmax of largest
      eigenvalue ev
  [V,D] = eig(F);
  ev = D(1,1); evvmax = V(:,1); %first ev
      and evvmax
  for i=2:4
        if D(i,i)>ev
              ev = D(i,i); evvmax = V(:,i);
                   %remember largest ev and
                   evvmax
        end
  end
  [E,P]=reser(X,Y,evvmax); %compute error
      and optimal P=UX
end
```

Listing 5: Computing the optimal predictor P and the error E.

```
function [E,P] = reser( X, Y, u )
  [n,m] = size(X);
  E=0; P=[];
  for k=1:m
        y = rotquat([0; X(k,:)],u);       %
            rotate the x vectors
        P=[P y(2:4)']; %store in P=UX
        E = E + norm( Y(:,k)~-~P(:,k) )$^\
            wedge$2; %sum of norms of
            differences
  end
  E = E/m;
end
```

The Matlab/Octave function `reser` shown in Table 5 shows how to compute the error E given by (4) and the optimal predictor $P = UX$. Note that using the largest eigenvalue of the matrix F directly, as proposed in [5], frequently results in complex values due to roundoff errors and negative values inside the root operation (see definition of e_q in [5]), which of course mathematically is impossible. Thus we recommend to directly compute E by using (4).

6. EVALUATION

To evaluate our approach of determining the spatial alignment of 3D acceleration sensor and derotating time series before doing comparisons we apply it to real world acceleration data. We use acceleration time series recorded pairwise by shaking two 3D accelerometers together and estimate if two devices were shaken together based on various similarity measures on their time series. In order to quantify the gain of derotating time series before comparing them we separately compare pairwise time series without and with derotation. Similarity between individual axes of two 3D accelerometers strongly depends on the spatial alignment of the accelerometers — therefore, comparing the original (arbitrarily aligned) axes directly with each other cannot be expected to yield useful results. Consequently, when not derotating time series, we apply the well known practice of computing and comparing the magnitude time series instead of utilizing individual axes. Before calculating these magnitudes we compensate for gravity by normalizing time series and subtracting their mean (to discard the influence of gravity) for each axis and device. To demonstrate that derotating time series is possible even with short recordings, we limit all time series to a duration of 2 s, which seems a compromise between distinguishability and usability for the user-mediated device pairing problem (cf. [17, 16].

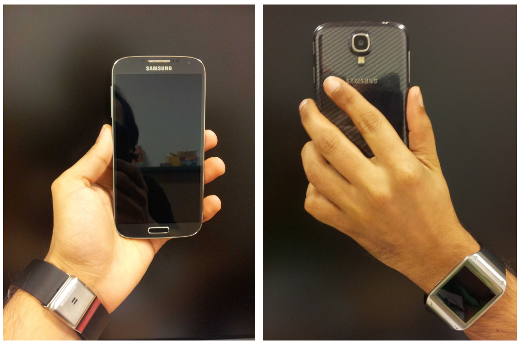
To compare two acceleration time series we utilize well known approaches – indicating either the amount of divergence (error) or similarity. As distance metrics indicating error we use: root mean squared error (RMSE), mean absolute deviation (MAE) and median absolute deviation (MEDIAN). As metrics indicating similarity we use correlation coefficients by Pearson (Product-Moment Correlation Coefficient [18]), Spearman (Rank Correlation Coefficient [19]) and Kendall (Tau Rank Correlation Coefficient [9]) as well as magnitude squared coherence, which has been used frequently on comparing acceleration time series in previous research [1, 4, 6, 7, 14, 17]. For coherence we apply parametrization as stated by Mayrhofer and Gellersen [17]. We expect coherence – representing the most sophisticated amongst the selected approaches – to yield better results than correlation coefficients, which we further expect to yield better results than the selected error based metrics.

6.1 Evaluation data

As source of acceleration time series we use the u'smile ShakeUnlock database[1] [7]. It contains pairwise 3D acceleration time series of two devices shaken together: a mobile phone held in the hand and a watch strapped to the wrist (Figure 1), shaken for about 10 s. In total, the database contains 29 participants shaking two devices 20 times, which results in 580 records of each two 3D acceleration time series. Acceleration has been recorded with 100 Hz across all devices. As we limit time series to a duration of 2 s we therefore only utilize 200 values per time series for comparison.

6.2 Time series derotation example

Figure 2 shows two sample magnitude time series from the u'smile ShakeUnlock database. The samples were originated by two devices actually shaken together. Axes have been gravity adjusted before calculating the magnitudes, and the time series have been limited to a duration of 2 s. Although magnitudes are not equal, their similarity is obvious: phasing is similar, though overall the amplitude seems to be higher for device 2.



(a) Front side (b) Rear side

Figure 1: Acceleration time series recording setup with the u'smile ShakeUnlock database [7].

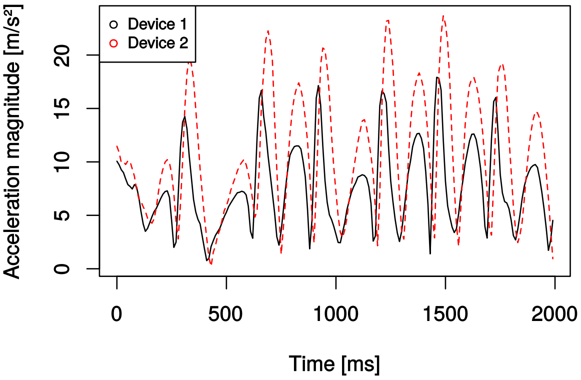
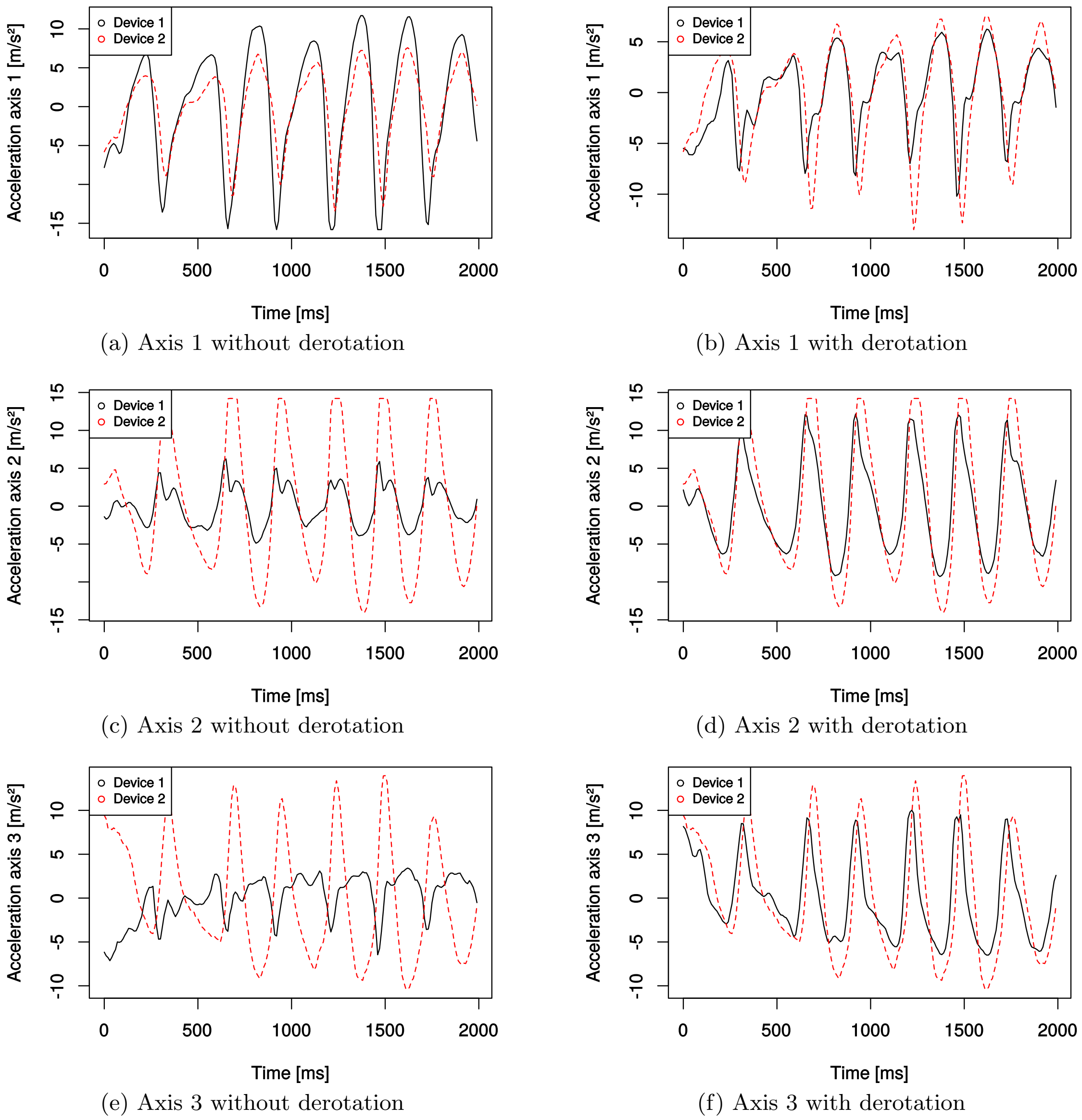


Figure 2: Sample acceleration time series magnitudes of device 1 and 2.

Looking at time series of individual axes for the same samples, similarity is not as obvious anymore (Figure 3). Although acceleration phasings and amplitudes are similar for axis 1, for axis 2 only phasings are obviously similar – for axis 3 there is no obviously visible similarity evident. After applying derotation (by rotating the 3D acceleration time series of device 1 according to the spatial alignment of device 2) similarity is obvious again for all axes.

Table 1 provides previously stated metrics for these two sample time series – for comparing magnitudes and individual axes, without and with applying derotation. As expected, comparing not-rotated time series of individual axes causes highest errors/least similarities. Comparing magnitudes causes smaller errors/higher similarities. Overall, smallest errors/highest similarities were achieved by comparing derotated, individual axes.

[1] Evaluation data fetched from `http://usmile.at/downloads`



(a) Axis 1 without derotation

(b) Axis 1 with derotation

(c) Axis 2 without derotation

(d) Axis 2 with derotation

(e) Axis 3 without derotation

(f) Axis 3 with derotation

Figure 3: Sample 3D acceleration time series axes without (a, c, e) and with derotation (b, d, f).

		Without derotation			With derotation		
Metric	Mag.	A1	A2	A3	A1	A2	A3
RMSE	5.65	4.25	6.71	8.39	3.24	4.16	4.49
MAE	4.44	3.23	5.69	7.34	2.28	3.52	3.67
MEDIAN	3.88	2.71	5.15	7.71	1.38	3.56	3.24
Pearson	0.56	0.87	0.87	-0.57	0.77	0.91	0.74
Kendall	0.44	0.76	0.69	-0.49	0.68	0.77	0.60
Spearman	0.61	0.92	0.88	-0.68	0.86	0.92	0.80
Coherence	0.80	0.61	0.56	0.60	0.63	0.65	0.64

Table 1: Similarity metrics of sample acceleration time series, for magnitudes (Mag.) and individual axes (A1-A3), without and with derotating axes before comparison.

6.3 Experimental setup

As the effect of increasing similarity between two 3D time series applies for correlated time series (devices shaken together) as well as for uncorrelated time series (devices not shaken together) an evaluation must include both correlated and uncorrelated time series. For this reason we determine if devices were shaken together by comparing all possible combinations of acceleration time series from the database. Each comparison results in a single, scalar metric value s. Applying a threshold t so that $\min(s) \leq t \leq \max(s)$ to s we obtain the true match rate (TMR, rate of devices correctly being identified as shaken together) and the true non match rate (TNMR, rate of devices correctly being identified as not shaken together). The false match rate (FMR, rate of devices incorrectly being identified as shaken together) is the inverse of the TNMR. For error based metric values (RMSE, MAE, MEDIAN), if $s < t$ time series are identified as shaken together – if $s \geq t$ time series are identified as not shaken together. Respectively, for similarity based metric values (correlation coefficient, coherence), if $s < t$ time series are identified as not shaken together – if $s \geq t$ time series are identified as shaken together. Comparison with ground truth (if the two time series have been recorded from devices actually shaken together) originates the TMR and TNMR (resp. FMR) used in the receiver operating characteristic (ROC) curves. To obtain the TMR we perform a pairwise comparison of all 580 pairs of acceleration time series from the database. To obtain the TNMR (resp. FMR) we use all $\frac{1160 \cdot 1159}{2} = 672220$ other possible pairwise comparisons of time series[2].

6.4 Results

Evaluation results clearly show that derotating and comparing individual axes of acceleration time series (Figure 4(a)) yields better results over computing and comparing their magnitudes (Figure 4(b)). This supports our hypothesis of derotating pairwise 3D acceleration time series before doing comparisons improving comparison results. Table 2 states the equal error rate $\text{EER} = 1 - \text{TMR} \simeq 1 - \text{TNMR}$ and the square root of the minimum squared error rate $\sqrt{\text{MSER}} = \sqrt{\min((1-\text{TMR})^2 + (1-\text{TNMR})^2)}$ for comparing time series based on magnitudes and on derotated, individual axes for all metrics[3].

	Without derotation		With derotation	
	EER	$\sqrt{\text{MSER}}$	EER	$\sqrt{\text{MSER}}$
RMSE	0.41	0.55	0.25	0.36
MAE	0.42	0.58	0.26	0.36
MEDIAN	0.46	0.63	0.27	0.38
Pearson	0.32	0.40	0.18	0.25
Kendall	0.32	0.40	0.18	0.26
Spearman	0.32	0.40	0.18	0.26
Coherence	0.18	0.24	0.16	0.22

Table 2: Axes derotation increasing similarity of axes from two sample 3D acceleration time series.

As expected, overall coherence based comparison yields best results, followed by correlation coefficients and less sophisticated error based metrics. Although results for error based metrics are close to random when based on magnitudes, there is a significant improvement when derotating and comparing individual axes instead. Interestingly, for correlation coefficient based metrics derotating and comparing single axes significantly decreases the FMR for high TMR. The close-to-flat area in the magnitude ROC curve indicates, that – for the data used in our evaluation – magnitude-correlation-coefficient based separation without significantly increasing FMR is possible for about 70-80% of samples. Separation of the remaining 20-30% is either erroneous – or causes a significant rise in FMR. This effect disappears when derotating and comparing individual axes.

7. CONCLUSIONS

We have contributed a method for determining relative spatial alignment of devices based on independently recorded acceleration time series during common movement. Using quaternions, our approach allows to *analytically* compute the optimal rotation between the respective reference systems with a run-time complexity of $O(N^2)$ for N samples. The significant advantage over heuristic approaches is that this method is guaranteed to provide the optimal rotation with deterministic run-time. We suggest that this approach is beneficial for all applications comparing acceleration (or other 3D sensors) time series that were recorded independently with potentially arbitrary and unknown alignment, and that it can be used on systems with limited computational resources such as mobile phones.

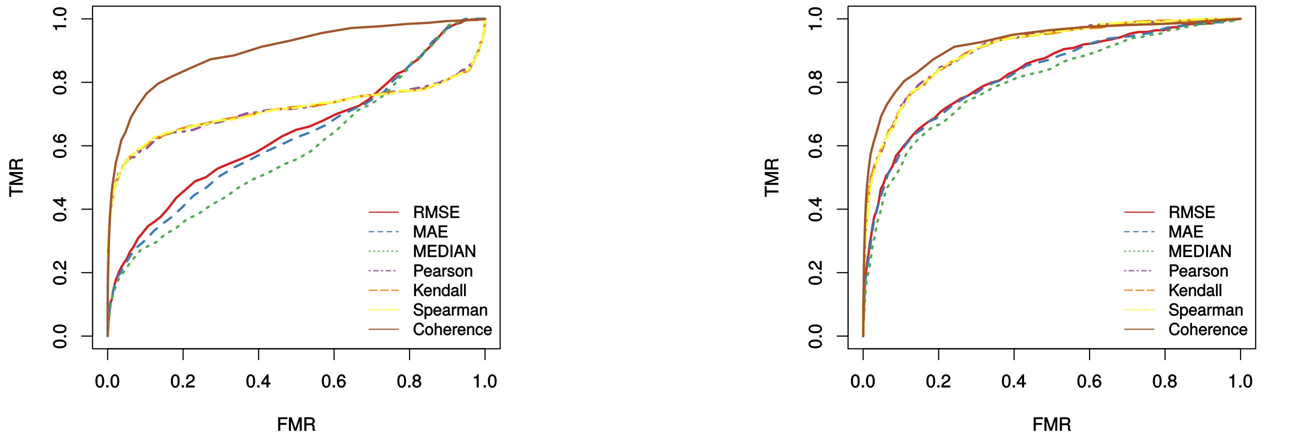
Using real world experimental data and coherence as the currently best performing distance metric for determining if two devices were shaken together, we see an improvement of about 11% in equal error rate by derotating the coordinate system of one of the devices before comparison. We note that this is the approach taken in previous research, relying on magnitude only and discarding angular information of all movement, and it still benefits from applying the method proposed in this paper.

We suspect that other methods for comparing 3D time series and using this additional information – which was previously impossible with arbitrarily rotated devices – can achieve significantly lower error rates. Our proposed analytical derotation method therefore opens new research questions for future work.

All Matlab/Octave scripts and data sets are available under the terms of the GNU Lesser General Public License (LGPL) at `http://ANONYMIZED`.

[2] Assuming that comparing time series is commutative, namely comparing time series A with B yields the same results as B with A – which applies to all our metrics.

[3] $\sqrt{\text{MSER}}$ represents the euclidean distance between the point TMR = TMR = 1 and the resulting TMR/TNMR closest to this point.



(a) Magnitude based comparisons

(b) Derotated individual axis based comparisons

Figure 4: Performance of deciding if devices have been shaken together using their acceleration time series, based on comparing magnitudes and derotated, individual time series.

Acknowledgments

This work has been carried out within the scope of *u'smile*, the Josef Ressel Centre for User-Friendly Secure Mobile Environments. We gratefully acknowledge support by the Christian Doppler Gesellschaft, A1 Telekom Austria AG, Drei-Banken-EDV GmbH, LG Nexera Business Solutions AG, and NXP Semiconductors Austria GmbH.

8. REFERENCES

[1] H. Ben-Pazi, H. Bergman, J. A. Goldberg, N. Giladi, D. Hansel, A. Reches, and E. S. Simon. Synchrony of rest tremor in multiple limbs in parkinson's disease: evidence for multiple oscillators. *Journal of Neural Transmission*, 108(3):287–296, 2001.

[2] D. Bichler, G. Stromberg, M. Huemer, and M. Löw. Key generation based on acceleration data of shaking processes. In *Proc. UbiComp 2007*, volume 4717 of *LNCS*, pages 304–317. Springer-Verlag, 2007.

[3] S. B. Choe and J. J. Faraway. Modeling head and hand orientation during motion using quaternions. Technical Report 2004-01-2179.

[4] C. T. Cornelius and D. F. Kotz. Recognizing whether sensors are on the same body. *Pervasive and Mobile Computing*, 8(6):822–836, 2012. Special Issue on Pervasive Healthcare.

[5] E. A. Coutsias, C. Seok, and K. A. Dill. Using quaternions to calculate rmsd. *Wiley InterScience*, July 2004.

[6] W. Dargie. Analysis of time and frequency domain features of accelerometer measurements. In *Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on*, pages 1–6, 2009.

[7] R. D. Findling, M. Muaaz, D. Hintze, and R. Mayrhofer. Shakeunlock: Securely unlock mobile devices by shaking them together. 2014. *unpublished*.

[8] B. Groza and R. Mayrhofer. SAPHE - simple accelerometer based wireless pairing with heuristic trees. In *Proc. MoMM 2012: 10th International Conference on Advances in Mobile Computing and Multimedia*, pages 161–168, New York, NY, USA, December 2012. ACM Press.

[9] M. Kendall. A new measure of rank correlation. *Biometrika*, 1938.

[10] D. Kirovski, M. Sinclair, and D. Wilson. The Martini Synch. Technical Report MSR-TR-2007-123, Microsoft Research, September 2007.

[11] J. B. Kuipers. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality.* Princeton Univers. Press; Auflage: Reprint, 2002.

[12] K. Kunze and P. Lukowicz. Dealing with sensor displacement in motion-based onbody activity recognition systems. In *Proc. Ubicomp 2008.*

[13] J. Lester, B. Hannaford, and G. Borriello. "Are you with me?" – Using accelerometers to determine if two devices are carried by the same person. In *Proc. Pervasive 2004*, pages 33–50. Springer-Verlag, 2004.

[14] J. Lester, B. Hannaford, and G. Borriello. ”are you with me?” - using accelerometers to determine if two devices are carried by the same person. In *Pervasive*, pages 33–50, 2004.

[15] R. Marin-Perianu, M. Marin-Perianu, P. Havinga, and H. Scholten. Movement-based group awareness with wireless sensor networks. In *Proc. Pervasive 2007*, pages 298–315. Springer-Verlag, 2007.

[16] R. Mayrhofer and H. Gellersen. Shake well before use: Authentication based on accelerometer data. In *Proc. Pervasive 2007: 5th International Conference on Pervasive Computing*, volume 4480 of *LNCS*, pages

144–161, Berlin, Heidelberg, Wien, May 2007. Springer-Verlag.

[17] R. Mayrhofer and H. Gellersen. Shake well before use: Intuitive and secure pairing of mobile devices. *IEEE Transactions on Mobile Computing*, 8(6):792–806, June 2009. revised and extended version of [16].

[18] K. Pearson. Note on regression and inheritance in the case of two parents. In *Proceedings of the Royal Society of London*, volume 58, pages 240–242, Jan. 1895.

[19] C. Spearman. The proof and measurement of association between two rings. *American Journal of Psychology*, 15:72–101, 1904.