Texture-based Tracking in mm-wave Images

Peter Salz[1], Gerd Reis[2], and Didier Stricker[2]

1 **University of Kaiserslautern**
Kaiserslautern, Germany
salz@rhrk.uni-kl.de
2 **Augmented Vision Group**
DFKI GmbH, Kaiserslautern, Germany

Abstract

Current tracking methods rely on color-, intensity-, and edge-based features to compute a description of an image region. These approaches are not well-suited for low-quality images such as mm-wave data from full-body scanners. In order to perform tracking in such challenging grayscale images, we propose several enhancements and extensions to the Visual Tracking Decomposition (VTD) by Kwon and Lee [6]. A novel region descriptor, which uses texture-based features, is presented and integrated into VTD. We improve VTD by adding a sophisticated weighting scheme for observations, better motion models, and a more realistic way for sampling and interaction. Our method not only outperforms VTD on mm-wave data but also has comparable results on normal-quality images. We are confident that our region descriptor can easily be extended to other kinds of features and applications such that tracking can be performed in a large variety of image data, especially low-resolution, low-illumination and noisy images.

1998 ACM Subject Classification I.4.8 Scene Analysis – Time varying imagery, Tracking

Keywords and phrases Visual Tracking Decomposition, low-quality images, texture features, mm-wave imagery

Digital Object Identifier 10.4230/OASIcs.VLUDS.2011.89

1 Motivation

To improve airport security, the detection of threats (especially non-metallic ones) attached to a person's body is very important. Full-body scanners that use mm-wave radiation are currently in development with the goal of automated threat detection. Smith Heimann GmbH provided us with several image sequences obtained from such scanners that show a person (carrying one or more threats) with raised arms performing a full rotation around the vertical axis. Our goal for the master's thesis by Salz [13] was to achieve an automated tracking of a threat through the sequence using a manually selected initial bounding box of the threat.

The grayscale images are very challenging with respect to the tracking method since they have a low resolution (with even smaller object sizes) and low contrast while several artefacts further reduce quality: Even between a very small number of frames we observe severe appearance and illumination changes, whole body parts are shadowed for some time, metal objects produce bright responses, and flares decrease the already small SNR.

Since these image sequences are confidential property of Smith Heimann GmbH, we are not able to provide any graphical descriptions. The sample images of Figure 1[1] should give a

[1] http://upload.wikimedia.org/wikipedia/commons/d/d8/Mmw_large.jpg

Proceedings of IRTG 1131 – Visualization of Large and Unstructured Data Sets Workshop 2011.
Editors: Christoph Garth, Ariane Middel, Hans Hagen; pp. 89–101
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
IRTG 1131

rough overview of the image quality, although no threats are shown and real-world scanner data are usually much noisier with less contrast and illumination.

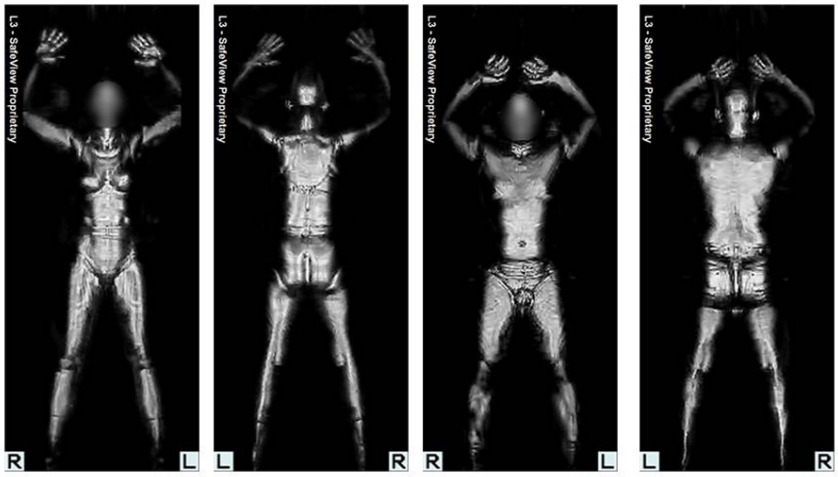


Figure 1 Sample mm-wave images of a female and a male person[1].

2 Overview

2.1 Contributions

We conducted a thorough analysis of several detection and tracking algorithms with respect to their applicability to mm-wave sequences (as is described in detail in [13]). Since we discovered several shortcomings of Visual Tracking Decomposition (VTD), we propose some enhancements (with respect to general tracking) and extensions (to apply it to other kinds of image data), such as a more sophisticated observation and motion model, better sampling and tracker interaction and integration of our own region descriptor.

This region descriptor is based on texture features as known from ultrasound texture discrimination and is computed using a patch decomposition of an image region, taking a carefully designed weighting scheme into account. We adapted the Förstner distance to compare sample and model covariance matrices computed from these patches in a robust way.

2.2 Results

As presented in Section 5, we achieved quite promising results on mm-wave sequences, especially compared to other approaches like VTD. When applied to the same data Kwon and Lee used for VTD, our method is comparable to it in terms of location error and sometimes even has better results.

In other low-resolution, low-light grayscale image data we expect our method to be superior to VTD as well, since the texture-based region descriptor is generally well suited for objects that are not easily represented by color and edge features.

2.3 Paper Organization

This paper is organized as follows: Section 3 introduces all related publications that were the basis for the proposed algorithm. Section 4 describes our algorithm in detail, presents all the sub-parts and gives an overview of the whole algorithm. In Section 5, some results are presented, including a report on mm-wave tracking and a comparison to Visual Tracking Decomposition on normal data. Finally, Section 6 concludes the paper, outlining our contributions and future work.

3 Related Work

3.1 Introduction

There are two dominant approaches for matching an image region to a template. In detection methods, the target object's position, scale, rotation, and other parameters can be almost arbitrary, i.e. the relation between the template and the target image is not known. This is especially helpful when dealing with images of the same scene, taken in different camera configurations, at a different time or in different circumstances (e.g. lighting). One of the major drawbacks is the global and usually exhaustive search in the target image. This is very expensive and often constrains the choice of a region descriptor, but more importantly, it might cause confusion of similar looking objects.

In video sequences on the other hand, tracking is performed in adjacent frames where only small changes both in appearance and in location are expected. This leads to the use of a motion model to predict the movement from frame to frame and an observation model to predict and cope with appearance changes.

Please refer to [13] for a much more extensive description and analysis of the presented methods.

3.2 Region Covariance

Region Covariance is a detection method proposed by Tuzel, Porikli and Meer [14] that uses the covariance matrix of pixel-based statistics as a region descriptor. It incorporates different features like position, color, and partial derivatives, but can easily be extended to other kinds of features. To make an exhaustive search in scale-space feasible, the covariance matrix of a region is computed using Integral Images, proposed by Viola and Jones [15]. This intermediate representation can be pre-computed for the whole image such that a covariance matrix for a region can be computed in constant time. To compare covariance matrices of a template and target region, the Förstner distance (Förstner and Moonen [4]) is used. This distance is based on generalized eigenvalues and assumes that all eigenvalues are positive, which is usually not valid in noisy, real-world data. Therefore, we propose an adapted Förstner distance computation that only considers a few dominant eigenvalues, as presented in Section 4.4.

3.3 Incremental Learning

In contrast to Region Covariance, Incremental Learning is a multi-template tracking method proposed by Ross et. al. [12] that maintains an appearance model based on previous observations. Their algorithm starts with a single template and incrementally updates the appearance model while down-weighting older observations with a forget factor. An intensity vector for all pixels in the image region is used as a region descriptor ($\mathbf{I}$), while the

appearance model consists of the eigenbasis U from $A = U\Sigma V^T$ with $A = [\mathbf{I}_1, ..., \mathbf{I}_n]$. Their update procedure incorporates new observations without re-computing the whole eigenbasis. A Gaussian motion model and a particle filter with the state consisting of translation, rotation, scale, aspect ratio and skew are used for tracking. See for example the tutorial by Arulampalam, Maskell and Gordon [1] for an overview of particle filters.

The method has some disadvantages, such as the high-dimensional feature vector that only includes raw intensity and requires a scaling of each target region to a common size. Furthermore, the Mahalanobis distance is used as the dominant distance measure for the observation model, utilizing a distance from the mean observation vector. This mean might not be very expressive, since the implicit assumption of a normal distribution of observation templates is not always valid in low-quality, noisy image data. The eigenbasis computation using Principal Component Analysis (PCA) allows for fast frame rates, since only a partial update of the covariance matrix is necessary for new observations. On the other hand, since we use a sparse variant of PCA (SPCA, see below), this is not feasible and a full covariance matrix computation is needed for each model update.

The incremental update procedure, the forget factor, and the subsequent implicit limitation of observation model size are valuable contributions to multi-template tracking and are also utilized in our method (see Section 4.3).

3.4 Visual Tracking Decomposition

Visual Tracking Decomposition (VTD) is a multi-template, particle-filter based tracking method proposed by Kwon and Lee [6]. They partition the observation model into different sub-models which are then utilized by basic trackers with different motion models. As state, the position and a single scale parameter are used, while the HSV representation (for grayscale images: intensity) and Sobel-based edge strength serve as features. To compute similarity of an observation to a sub-model, the Diffusion Distance by Ling and Okada [7] is used. Motion models consist of Gaussians with different variances (representing small and abrupt displacements between frames) and the partitioning of the observation model is computed by Sparse Principal Component Analysis (SPCA, d'Aspremont et. al. [3]) of the (non-centered) covariance matrix. SPCA produces sub-models that capture the most severe appearance changes, have a compact representation, and are almost complementary.

The eight basic trackers (with four observation and two motion models) generate Markov chains of samples using the Metropolis Hastings algorithm [5]. Interactive Markov Chain Monte Carlo (IMCMC) by Campillo et. al. [2] is used to fuse those basic trackers. One tracker proposes a state and the others decide whether to leap to this state or maintain their own Markov chain.

The authors report that their method can cope with severe appearance changes (because of the multi-template model), occlusion and illumination changes (robust features), and abrupt motion (different motion models). However, variance parameters for the motion models and the scaling parameter are adapted to the specific image sequence and the implementation deviates from the description in the paper. First, the motion model is centered at the original object position, not the current one, which is helpful for most of the sample sequences where the camera follows the object of interest. Second, basic trackers propose new states in each iteration, although in the paper this sampling is only done with a certain probability or an interaction step is performed. Furthermore, VTD has several other drawbacks that are addressed in our method. The features are not applicable to low-quality images. The old position of a basic tracker is ignored when sampling new states and Metropolis-Hastings sampling yields an acceptance rate close to 100% (while theoretically it should be around

23%, see Roberts et. al. [11]). The entries of sparse principal components are not utilized to weight their influence, although their variance contribution differs significantly.

3.5 Texture Features

In most tracking applications, a combination of color or intensity and edge information is used to compute feature vectors of an image region. For low-quality, noisy and colorless images, these region descriptors perform very poorly. In the field of ultrasound texture classification, another set of features has been proposed that relies on intensity and several intensity transformations (gradient magnitude, difference to mean, horizontal and vertical residuals), as described by Muzzolini et. al. [10] and Liu and Jernigan [9].

The following set of features both from the spatial as well as from the frequency domain has been chosen for our method: The Kolmogorov-Smirnov distance between the cumulative distributions (approximated by histograms) of each target patch and a representative sample (which is known to belong to the texture class of interest) and the standard deviation of each patch are spatial features. Frequency features include the energy at the major peak of the normalized power spectrum (given by the Fourier transform), the Laplacian of major and secondary peak, relative orientation of major and secondary peaks, isotropy of the power spectrum, and relative energy and entropy of inner regions of the power spectrum.

3.6 Sparse Principal Component Analysis

One disadvantage of Principal Component Analysis (PCA) is the non-sparsity of entries. Just thresholding (such that small entries are ignored) is not sufficient for dimensionality reduction. Therefore, a variant of PCA is used, which captures most of the variance in the data, but with a desired sparsity of principal components. It is called Sparse Principal Component Analysis (SPCA). Please refer to [13] for a detailed description.

In our method, the sparse principal components (SPCs) are computed using an iterative elimination strategy (Wang and Wu [16]) with project deflation to eliminate the influence of a pseudo-eigenvector. A single parameter controls the percentage of variance that this SPC should preserve.

4 Texture-based Tracking using Visual Tracking Decomposition

4.1 Model Representation

The tracking result for a single frame consists of a state and the corresponding observation or feature vector. In the original VTD approach, only a single scale parameter was used, but in many applications an object might change its scale in both dimensions, so we propose to incorporate a total of four state variables: x,y position of the region center and x,y scaling parameters for the scale change with respect to the first template. VTD also has a fixed observation model size of ten templates, while older templates are discarded and all current templates are treated equally, relying on SPCA to sort out features and time steps that do not capture important appearance changes. Instead, we make use of the forget factor, introduced in [12], to down-weight older samples and thus implicitly restrict the template size.

Since the observation model is supposed to capture appearance changes, a representative sample has to be chosen during model update (using the terminology introduced in [10]). In our approach, a manually selected template is initially available, which serves as the representative until the first model update. This representative is supposed to replace the

model mean, since it is a true part of the model (describing the object best in a certain point in time) and thus is more useful for computing the variance in the template dataset. The representative is described in Section 4.3.

The motion model is very similar to the one in VTD, although some of its shortcomings have been addressed. As proposal density functions for the Metropolis-Hastings algorithm, Gaussian functions with different parameters have been commonly used and are easily sampled. In an initialization phase, basic trackers are spread out from the current location. If they encounter a better state than the old one, they move to this state and continue sampling from there. In contrast to VTD, our motion models try to predict the movement in the current frame based on previously detected movements. The first motion model is centered at the current location of the basic tracker and has a location variance that depends on the absolute difference of the current and previous position (the larger this difference, the larger the variance). The second motion model assumes that an object continues its movement from the previous frame, so the mean of the Gaussian is shifted in that direction, using the same variances as the first motion model.

The eight basic trackers are constructed from combinations of those two motion models and the four basic observation models.

4.2 Feature Computation

Let R be an image region of size $W \times H$. In VTD, the different features were computed as a matrix of all features and pixels. For our texture-based approach the whole region is not discriminative enough since the texture features are supposed to be constructed from small patches. Therefore, we partition the region into five smaller, overlapping patches. These patches are evenly sized, and four of them fill the whole region while overlapping each other slightly, thereby ensuring that structures near patch borders are not omitted. The fifth patch is placed in the center of the region, overlaps with all other patches, and usually does not contain any irritating non-object pixels from the borders of the region.

Since the center patch is most likely to contain only parts of the object of interest, it is assigned a dominant weight of 0.5, while the remaining four patches get a weight of 0.125 each. For the patch decomposition we chose an overlap parameter $\omega \in [0, 1]$, which we set to $\omega = 0.1$. The patch width (for all patches) is $W_\omega = [0.5 \cdot (1 + \omega) \cdot W]$, with $[\cdot]$ being the round operator, and the patch height is computed accordingly.

For each of these five patches, the $d = 21$ features (see Section 3.5) are computed, resulting in a 21×5 feature matrix $F(R)$ for the region R. Note that this matrix is independent from the region size, so no scaling is necessary compared to the Incremental Learning approach in [12].

4.3 Model Update using SPCA

Observation model size

The observation model is initialized with the manually selected reference template, so for the first few frames our method is equal to a single template region matching. After m frames, we therefore gathered $m + 1$ templates from which the new observation model is computed. With a forget factor $f \in [0, 1]$ and a total number of templates t, the new effective database size after the update becomes $t \leftarrow f(t - m) + m$. In VTD, m is set to two and $f = 1$, while the database size is restricted to ten templates. In Incremental Learning, $m = 5$ and $f = 0.95$ with an effective database size of 100 templates. For our approach we set $m = 3$ and $f = 0.8$ because of the severe appearance changes in mm-wave sequences, resulting in a maximum

database size of $\frac{m}{1-f} = 15$. Therefore, if the database size is larger than the effective size, the oldest templates are discarded (because they do not contribute to the model anymore) and the weights of remaining templates are re-normalized.

Representative sample

At the beginning of each model update, a new representative template is computed as follows:

1. Find median m_k^p for each feature $1 \leq k \leq d$ and patch $1 \leq p \leq 5$, resulting in a $d \times 5$ matrix of median values.
2. Compute the $d \times d$ covariance matrix for this median and for each template from their patches.
3. Compute Förstner distance between covariance matrices of median and each template.
4. The template with smallest distance to the median is chosen as new representative F_{Rep}.

This representative template is a true part of the model which describes the object best at a certain time step, and it also has a minimal distance to the median, so any template with a large variance with respect to this representative describes a certain appearance change of the object.

Template covariance matrix

Each template has a feature matrix $F_l, l = 1, ..., t$ and a weight w_l (the maximum a-posteriori value from the tracking process), which gets down-weighted by powers of the forget factor. For $l = t-1, ..., t-m$ for example, $f_l = f^0 = 1$, while the exponent is increased by one for every m weights.

The template covariance matrix C_M is constructed from the $d \times 5t$ data matrix $D_M[k, 5 \cdot l + p] = \sqrt{w_l \cdot f_l}\,(F_l[k,p] - F_{\text{Rep}}[k,p])$ with $1 \leq k \leq d$ features, $1 \leq l \leq t$ time steps, $1 \leq p \leq 5$ patches, and F_l the l-th template's feature matrix. D_M is normalized by $\frac{1}{\sqrt{\sum_l f_l \cdot w_l}}$, resulting in the template covariance matrix

$$C_M = \frac{1}{1 - \frac{\sum_l (w_l f_l)^2}{\left(5 \cdot \sum_l w_l f_l\right)^2}} D_M \cdot D_M^T. \tag{1}$$

SPCA

The purpose of SPCA is to reduce the number of features and time steps that contribute to a submodel to only those that explain severe appearance changes (with respect to the representative), and to split the observation model into different submodels for each basic tracker. To compute sparse principal components from the template covariance matrix, we use the Iterative Elimination algorithm by Wang and Wu [16] since it is easy to compute and is controlled by a single parameter that controls the amount of variance a sparse principal component (SPC) should contain.

For the $r = 4$ observation models, we need the first four SPCs of C_M, where the first SPC captures 40% percent of the whole variance and the remaining SPCs 24%, 9.6% and 3.84%, respectively, so 77.54% of the total variance is captured by the basic observation models. The corresponding eigenvalues serve as weights for each SPC (normalized such that they sum up to one) and the first SPC u_1 is obtained from the full covariance matrix C_M. For the other SPCs u_c, a projection deflation $C_M \leftarrow \left(I - u_{c-1}u_{c-1}^T\right) C_M \left(I - u_{c-1}u_{c-1}^T\right)$ is used to remove the influence of other pseudo-eigenvectors u_{c-1}.

Feature reduction

For each basic observation model $1 \leq c \leq r$, we obtained a sparse principal component u_c. If its k-th entry is considerably larger than zero (where $1 \leq k \leq d$), feature k is added to the sub-model c with weight $u_c[k]$. Sometimes only one feature is dominant with a weight close to one. Unfortunately, in that case the time step selection described in the next section does not work, so we add the next feature $k + 1 \bmod d$ with weight 0.1.

After the feature selection, each submodel c contains d_c features, while some features may contribute to more than one model since the SPCs are not completely orthogonal (in contrast to true principal components).

Time step selection

To further reduce the complexity of the basic observation models, we use only those time steps that exhibit a large variance with respect to the representative. This differs from the VTD approach where feature reduction and time step selection are performed in a single application of SPCA. In contrast, due to the complex but more realistic weighting scheme in our method, we apply an additional SPCA to select relevant time steps.

For each remaining feature of a basic observation model we compute the $t \times t$ covariance matrix from the five patches of templates. From SPCA we obtain the first SPC that captures 75% of variance in all time steps and store it in a $t \times d_c$ weight matrix T_c.

For each time step we decide to keep or discard it:

1. $t \times 5$ data matrix D_t for each feature $1 \leq k \leq d_c$, weighted as above, each row $1 \leq l \leq t$ is multiplied by $w_l \cdot f_l$ (w_l is the weight of the time step, f_l influence of the forget factor) $\Rightarrow t \times t$ covariance matrix $C_t \Rightarrow$ SPC $u_k \Rightarrow k$-th column of T_c.
2. For each row of T_c take the average (weighted with feature weights from feature reduction), if larger than 0.15, add time step to basic observation model c with weight w_l multiplied by weighted average.
3. Also add the representative time step to each model such that each basic model contains a representative observation and appearance changes.

Model update

For each basic observation model c we now have d_c features and t_c time steps, so all weights are normalized to sum up to one and the normalized eigenvalue from the c-th SPCA is the weight of that model. This leads to a covariance matrix C_c for that basic model, which is positive semi-definite, so in practice not all singular values are non-zero. For the sample weight computation below, we eventually need the inverse of the covariance matrix, so using a Singular Value Decomposition $C_c = U\Sigma V^T$ we construct a pseudo-inverse $C_c^+ = V\Sigma'^T U^T$, where Σ' contains the reciprocal of each diagonal element that is larger than some $\epsilon > 0$. This pseudo-inverse is stored in the observation model since it is computed only during the model update but is used heavily during the sample weight computation.

4.4 Sample Weight Computation

Förstner distance

To determine the confidence that a region contains the object of interest (i.e. belongs to the observation model) we need a weight $w \in [0, 1]$ that is proportional to some distance that measures similarity between a sample and the model. For VTD the Diffusion Distance [7] was

chosen that compares histograms of the feature vectors. Our region representation instead consists of five feature vectors, one for each patch, that are stored in a feature matrix. Since these features have very different ranges (their theoretical bounds are known, for example intensity is in $[0, 255]$, but practical upper bounds are difficult to define), a normalization is applied based on empirical bounds. Note that sometimes a patch has a very low mean intensity (i.e. is almost black) such that the feature computation would not be robust. In such a case we discard the sample immediately by setting the weight to zero.

Each basic observation model c has t_c templates, each with the same d_c features, including the representative. The covariance matrix of feature matrix F for a sample region is computed using the patch-specific weights w_p (0.5 for the central patch and 0.125 for the others) and the mean (for feature k: $\mu_k = \frac{1}{5}\sum_{p=1}^{5} w_p \cdot F[k,p]$) from the data matrix $D[k,p] = \sqrt{w_p}(F[k,p] - \mu_k)$, resulting in $C_S = \dfrac{1}{1 - \sum_{p=1}^{5} w_p^2} D \cdot D^T$ (of size $d_c \times d_c$).

For each template $1 \leq l \leq t_c$ the covariance matrix C_l has been precomputed at the model update, so the Förstner distance is $FD(C_S, C_l) = \sqrt{\sum_{i=1}^{\dim_{C_S}} \log^2 \lambda_i}$, where $\dim_{C_S}$ is the number of eigenvalues of C_S that are larger than $\epsilon = 10^{-6}$ and λ_i is the i-th eigenvalue of matrix $C = C_S \cdot C_l^+$. By restricting the distance computation to only the $\dim_{C_S}$ largest eigenvalues we improve robustness since a lot of the smaller eigenvalues are very close to zero and therefore distort the distance function.

Let w_l be the weight from the SPCA, then the overall distance of a sample region to the model is the weighted average $d = \sum_{t=1}^{t_c} w_l \cdot FD(C_S, C_l)$.

Weight map

To transform the distance into a meaningful and expressive weight $w \in [0, 1]$, a weight map needs to be designed. In VTD, a simple exponential function was used to map the Diffusion Distance to a weight. In our low-quality images and with our specific features and distance, we observed that weights decrease too fast to very small values when using an exponential function. We also noted that there is a sharp boundary between samples that should be accepted or rejected. We therefore decided to use a sigmoid function since it should only take positive values, its maximum value should be at zero (because a distance of zero is a perfect match), for values larger than zero we expect a decrease in weights with a sharp transition from large to small weights.

This results in the following weight map with empirically tuned parameters:

$$w(d) = \frac{1}{1 + e^{4(d-2)}} \tag{2}$$

4.5 Tracker Interaction

The $r \cdot s = 8$ basic trackers independently produce samples from the same distribution. The interaction scheme used here is similar to VTD, but with a lower acceptance rate of samples, a more realistic spread of initial samples, a better state evaluation of other trackers, and a better weighting of a tracker's contribution.

In each iteration the interaction step is performed with probability α, which is initially set

to one and linearly decreases to 0.5. The sum of weights of all basic trackers is $s_w = \sum_{i=1}^{rs} \frac{1}{s}\lambda_i w_i$, where λ_i is the normalized eigenvalue (i.e. the significance of contribution) and w_i is the weight a tracker has at the current state.

For each basic tracker $1 \leq o \leq rs$ we have $p_o = \frac{1}{s}\lambda_o w_o \frac{1}{s_w}$, which, in the original VTD approach, is the probability that a proposed state is accepted by tracker c. It is not clear, however, whether this will indeed improve the state of this tracker. Therefore, we decided to compute the weight of tracker c at the state proposed by tracker o with probability p_o and accept it only if the weight is larger than the current one, thereby guaranteeing that c actually improves its state.

We also replace the Metropolis-Hastings sampling algorithm with its multiple-try variant [8] to increase confidence in a proposed sample by drawing more intermediate samples. This requires less iterations (indeed we reduce VTD's 100 iterations to only 20, because the maximum a-posteriori is usually found very fast) and also produces better samples.

First, ten samples y_i are drawn from the motion model (which serves as proposal density function) associated with basic tracker c and compute the weight w_{y_i}. The samples are sorted by their weight, starting with the largest one, and one sample $y = y_i$ is selected with probability w_{y_i}. Then we center the motion model at y, draw nine additional samples x_j and set x_{10} to the current state of c. We compute weights w_{x_j} and the acceptance ratio $\gamma = \frac{\sum_{i=1}^{10} w_{y_i}}{\sum_{j=1}^{10} w_{x_j}}$. The state y is accepted with probability γ and y is added to the list of accepted samples. The accepted sample with largest weight is then selected as the maximum a-posteriori (MAP) estimate. This weight is added to the list of weights of all templates, which are then re-normalized.

Note that the acceptance rate for a Gaussian proposal density function should theoretically be around 23% (compare Roberts, Gelman and Gilks [11]). For VTD we observed acceptance rates of more than 95%, while our method mostly achieves acceptance rates between 17% and 40%.

4.6 Algorithm Summary

For every frame of the sequence:

1. Compute global intensity transforms.
2. After every 3 frames update observation model using SPCA.
3. Perform initial sampling using extended motion model.
4. Repeat for each iteration:
   - Tracker interaction with probability $\alpha \in [0.5, 1]$.
   - Reduce α linearly.
   - State sampling using Multiple Try Metropolis-Hastings algorithm.
5. Select maximum a-posteriori estimate from accepted states.

5 Results

Application to mm-wave data

As mentioned before, we cannot provide any graphical examples for mm-wave data, but we achieved promising results in our experiments, especially in comparison to the original VTD method which only used intensity and edge strength as features. To our knowledge, no other method is capable of dealing with this kind of challenging data.

Figure 2 shows a comparison of absolute pixel errors of the bounding box center with respect to a manually selected ground truth.

Note that in mm-wave sequences threats are only visible for a few frames and severe appearance and illumination changes occur even between a small number of frames. Additionally, threats might be subject to imaging artefacts and their appearances often do not deviate much from surrounding regions.

For more evaluations of tracking in mm-wave sequences, please refer to [13].

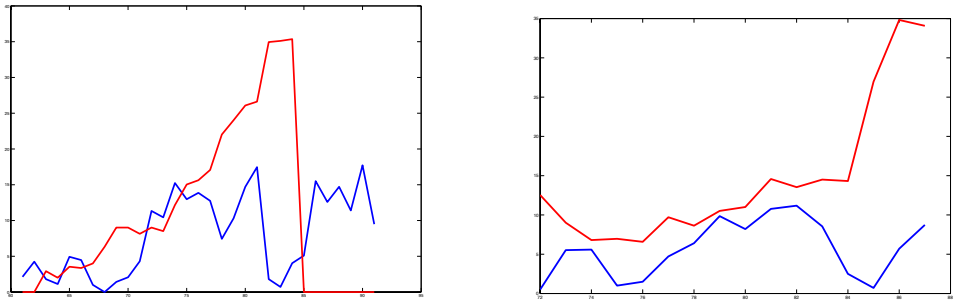
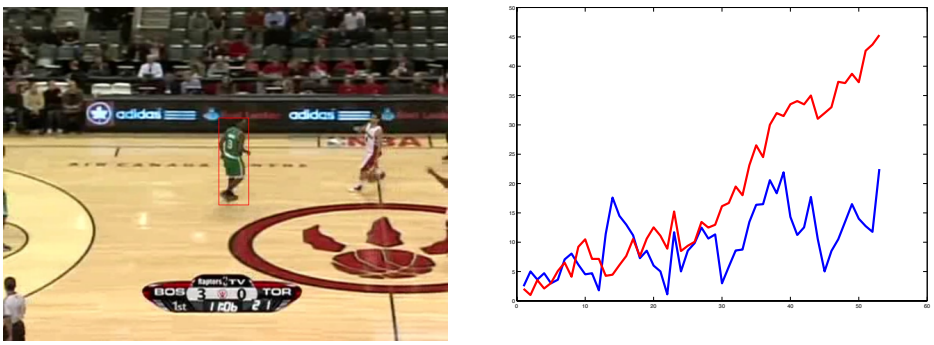


Figure 2 Results for two mm-wave sequences. The blue graph shows the errors of the presented method, while the red graph represents the VTD results. Pixel errors are on the vertical axis, frame numbers on the horizontal axis.

Comparison to VTD on standard data

We also compared our method to VTD using higher-quality images that Kwon and Lee used themselves. It turned out that our method mostly had similar results, sometimes even outperforming VTD, although with a lower framerate. This is depicted in Figure 3.



(a) A sample image

(b) As above, blue represents the presented method and red the VTD.

Figure 3 A sample sequence of real-world data. Note that in this case, our method actually outperforms the VTD.

6 Conclusion

6.1 Summary and Contributions

We presented an extension to the framework Visual Tracking Decomposition [6] that improves some of its shortcomings and enables tracking in low-quality images, especially mm-wave data from full-body scanners. We proposed a new region descriptor that is based on features introduced in ultrasound texture discrimination [10] and applied it successfully to mm-wave data and color images. Our method uses the Förstner distance [4] for region comparison, improves the multi-template observation model of VTD using a forget factor [12] and a more sophisticated weighting scheme, incorporates a better motion model and sampling method [8] for particle filtering, and replaces the model mean with the template closest to the model median.

In comparison with VTD our method is superior on mm-wave data (low-quality and noisy grayscale images), while it has similar results for data with color and strong edges.

6.2 Future Work

The feature set we have chosen consists of 21 features both from the spatial and the frequency domain. A comparative study is necessary to evaluate the usefulness of each feature in order to reduce the number of features to a minimum while keeping tracking quality. Furthermore, feature computation is quite expensive since most computations need to be done locally and more than 16000 Fourier transforms are necessary for each frame. An optimized and parallelized way of feature computation will improve frame rates significantly, although real-time results are not expected to be feasible. Finally, we would like to extend our method to track several objects simultaneously and also incorporate color features to improve tracking results in colored image data.

Since the presented tracking algorithm works well on mm-wave images, a very challenging kind of data that to our knowledge nobody ever dealt with, we definitely expect it to be well-suited for similar applications. Therefore, we plan to test and tune our method to a variety of different imaging techniques, such as infrared imaging and especially ultrasound data. We adapted the texture features from the ultrasound imaging domain, so our region descriptor should be capable of tracking objects or anatomical structures.

7 Acknowledgements

We would like to thank Smith Heimann GmbH for providing us with mm-wave data and discussing our research. We also thank Junseok Kwon and Kyoung Mu Lee for providing us with the source code of their VTD framework.

This work was partially funded by the German Research Foundation's International Research Training Group (IRTG) "Visualization of Large and Unstructured Data Sets, Applications in Geospatial Planning, Modeling and Engineering" and the German Research Center for Artificial Intelligence (DFKI GmbH).

References

1 M. Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

2 Fabien Campillo, Rivo Rakotozafy, and Vivien Rossi. Parallel and interacting markov chain monte carlo algorithm. *Mathematics and Computers in Simulation*, 79(12):3424–3433, 2009.

3 Alexandre d'Aspremont, Laurent El Ghaoui, Michael I. Jordan, and Gert R. G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. *SIAM Rev.*, 49:434–448, July 2007.

4 W Förstner and B Moonen. A metric for covariance matrices. *Ratio*, 66:113–128, 1999.

5 W.K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

6 Junseok Kwon and Kyoung Mu Lee. Visual tracking decomposition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010)*, 2010.

7 H. Ling and K. Okada. Diffusion distance for histogram comparison. *International Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, 2006.

8 Liang Faming Liu, Jun S. and Wing Hung Wong. The multiple-try method and local optimization in metropolis sampling. *Journal of the American Statistical Association*, 95(449):121–134, March 2000.

9 Song-Sheng Liu and M.E. Jernigan. Texture analysis and discrimination in additive noise. *Comput. Vision Graph. Image Process.*, 49:52–67, January 1990.

10 Russell Muzzolini, Yee-Hong Yang, and Roger Pierson. Texture characterization using robust statistics. *Pattern Recognition*, 27(1):119 – 134, 1994.

11 G.O. Roberts, A. Gelman, and W.R. Gilks. Weak convergence and optimal scaling of random walk metropolis algorithms. *Ann. Appl. Probab.*, 7(1):110–120, 1997.

12 David A. Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. 2008.

13 Peter Salz. Automated tracking of threats in imagery from mm-wave scanners. Master's thesis, University of Kaiserslautern, Germany, 2011.

14 Oncel Tuzel, Fatih Porikli, and Peter Meer. Region covariance: A fast descriptor for detection and classification. In *Proc. 9th European Conf. on Computer Vision*, pages 589–600, 2006.

15 Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, 1:511–518, 2001.

16 Yang Wang and Qiang Wu. Sparse pca by iterative elimination algorithm. *Accepted in Advances in Computational Math.*, 2010.