Training Data Selection and Update Strategies for Airborne Post-Doppler STAP

André Barros Cardoso da Silva, Stefan V. Baumgartner, Member, IEEE, and Gerhard Krieger, Fellow, IEEE

Abstract—Space–time adaptive processing (STAP) of multichannel radar data is an established and powerful method for detecting ground moving targets, as well as for estimating their geographical positions and line-of-sight velocities. Crucial steps for practical applications are: 1) the appropriate and automatic selection of the training data and 2) the periodic update of these data to take into account the change of the clutter statistics over space and time. Improper training data and contamination by moving target signals may result in a decreased clutter suppression performance, an incorrect constant false alarm rate threshold, and target cancelation by self-whitening. In this paper, two conventional and two novel methods for training data selection are evaluated and compared using real four-channel X-band radar data acquired with DLR’s airborne sensor F-SAR. In addition, a module for rejecting potential moving target signals and strong scatterers from the training data is proposed and discussed. All methods are evaluated for a conventional post-Doppler (PD) STAP processor and for a particular PD STAP that uses an a priori known road map.

Index Terms—Airborne radar, clutter statistics change, ground moving target indication (GMTI), radar applications, radar detection, radar signal processing, road vehicle detection, synthetic aperture radar (SAR), traffic monitoring.

I. INTRODUCTION

In space–time adaptive processing (STAP) algorithms, the clutter suppression is carried out by applying the inverse of the clutter covariance matrix (CCM). Since the clutter spectral properties are rarely known a priori, the CCM generally has to be estimated by using appropriate training data. The estimated CCM (and therefore the selected training data) influence the achievable clutter suppression performance and the statistics of the clutter suppressed data. However, only these statistics are used for fitting a clutter model and deriving the detection threshold for achieving a constant false alarm rate (CFAR). Thus, a contamination of the training data with strong discrete scatterers and interfering moving target signals may lead to an improper CFAR threshold setting and target self-whitening. Therefore, the training data selection plays an important role for an effective application of STAP [1]–[4].

An interesting literature review about training data selection methods is presented in [5]. However, several of the discussed algorithms are time-consuming (e.g., designed for joint-domain STAP, which requires more sample support and processing time) or may require a massively complex series of decisions to be made in real time, which is especially the case for knowledge-aided (KA) algorithms.

For instance, the KA parametric covariance estimation (KAPE) approach presented in [6] blends both a priori knowledge information and measured observations in order to mitigate the impact of heterogeneous clutter on space–time detection. Rather than estimating the whole CCM, the KAPE approach estimates the parameters of a CCM model for each individual range bin of interest. Although this approach is robust for STAP applications, it has unique computational demands to carry out the CCM reconstruction. Furthermore, it is pointed out that: 1) the accuracy of the modeled CCM determines the detection performance potential of KAPE and 2) the knowledge of the array manifold is critical for the clutter cancelation capability of KAPE. An advance of this framework is presented in [7] and is known as enhanced KAPE. This new approach improves the KAPE’s susceptibility to array errors by applying a new iterative calibration technique. Besides, the computational burden of KAPE is also improved by circumventing the need for direct inversion of data matrices for each range bin.

More recently, a KA algorithm showed that the generalized inner product (GIP) test can be applied in the space-Doppler domain in order to remove nonhomogeneous training data from the CCM estimation [8]. This approach requires the terrain database for obtaining the statistical properties of the clutter for each region of interest.

However, especially when real-time processing is desired, the use of KA algorithms (e.g., those presented in [6]–[8]) would further increase the complexity of STAP.

This paper presents an evaluation of four training data selection algorithms that can be combined with a module that rejects potential moving target signals and strong scatterers. The algorithms are applied on two ground moving target indication (GMTI) processors: the conventional post-Doppler (PD) STAP and a particular PD STAP that uses an a priori road map [9]. The main goal is to improve the clutter suppression capability and, thus, to increase the number of true detections.

The remainder of this paper is organized as follows. Section II introduces the PD STAP framework and the heterogeneous clutter model used for the evaluation of the
training data selection algorithms. Section III discusses some real-world effects that occur especially in airborne STAP. Section IV presents the training data selection algorithms. Section V presents an evaluation of these algorithms using real four-channel X-band data acquired by DLR’s airborne sensor F-SAR [10], [11]. Our discussion concludes in Section VI.

II. PD STAP AND CLUTTER MODEL

The PD STAP is a reduced rank algorithm that requires less sample support and less processing effort than the classical joint domain STAP [1]. The multichannel PD STAP processing is mainly carried out in range-Doppler domain. Just like other STAP algorithms, the PD STAP is able to perform clutter suppression, moving target detection, direction of arrival (DOA), and line-of-sight velocity estimation.

In practice, the application of PD STAP involves at least the following steps:

1) calibration or balancing of the multichannel data;
2) partitioning of the data along azimuth into small coherent processing intervals (CPIs);
3) successive azimuth FFT (fast Fourier transform) of all CPIs;
4) estimation of the CCM \( R_W \) (commonly by using appropriate training data);
5) computation of the test statistics \( \mathbf{T} \) for the broadside direction;
6) clutter model fit on \( \mathbf{T} \) and determination of the CFAR detection threshold;
7) target detection by applying the CFAR threshold on the test statistics \( \mathbf{T} \);
8) DOA-angle and line-of-sight velocity estimation for each detected target;
9) geocoding of the detected targets.

Sections II-A–II-C introduce the moving target signal model, the principle of the PD STAP algorithm, and the applied heterogeneous clutter model.

A. Signal Model and STAP Overview

The multichannel signal model for the PD STAP is given by [12]

\[
s(u(t)) = ae^{-j\frac{2\pi}{\lambda} R(t)} D_r(u(t))
\]

where \( a \) denotes the complex amplitude of the reflectivity of the scatterer, \( \lambda \) is the radar wavelength, \( M \) is the number of receive channels, \( R(t) \) denotes the range to the antenna array center, \( D_r(u(t)) \) and \( D_{r,m}(u(t)) \) denote the complex transmit and receive antenna characteristics of the \( m \)-th channel, \( x_m \) corresponds to the antenna phase center position in azimuth direction with respect to the array origin, \( d(u) \) is the DOA (or beamforming) vector, \( u(t) = \cos(\Psi_{\text{DOA},az}) \) is the directional cosine, and \( \Psi_{\text{DOA},az} \) is the DOA-angle of the target measured with respect to the azimuth direction, as shown in Fig. 1. In Fig. 1, \( R \) is the slant range of the target.

The directional cosine can also be expressed in terms of the Doppler frequency and the line-of-sight velocity of the target, according to

\[
u(t, f_a, v_r) = \cos(\Psi_{\text{DOA},az}) = \frac{\lambda f_a}{2v_p} + \frac{v_r}{v_p} \tag{2}\]

where \( f_a \) is the Doppler frequency, \( v_r \) is the line-of-sight velocity of the target, and \( v_p \) is the velocity of the platform.

The CCM can be estimated empirically from the data by applying the sample matrix inverse method [13], which is the basis for most modern STAP algorithms (note that other CCM estimation strategies are also possible [14])

\[
\hat{R}_W(f_a) = \frac{1}{K} \sum_{k=1}^{K} z(r_k, f_a)z^H(r_k, f_a), \quad \hat{R}_W(f_a) \in \mathbb{C}^{M \times M} \tag{3}\]

where \( K \) is the number of range bins used for averaging, \((\cdot)^H\) is the Hermitian operator, and \( z(r_k, f_a) \) denotes the multichannel data generally composed of [3]

\[
z = s + c + n + j \tag{4}\]

where \( s \) denotes the vector of the target signal given in (1), \( c \) is the clutter, \( n \) is the noise, and \( j \) is a possible jammer signal.

For CCM estimation, it is important to note that the multichannel vectors \( z \) should be free of strong discrete scatterers and moving target signals. This condition has to be ensured by a proper training data selection algorithm, presented and discussed in Section IV.

The moving target detection is carried out by applying the well-known Adaptive Matched Filter test [15]

\[
\mathbf{T}(r_k, f_a) = \frac{|d^H(u_t, f_a)\hat{R}_W^{-1}(f_a)z(r_k, f_a)|^2}{d^H(u_t, f_a)\hat{R}_W^{-1}(f_a)d(u_t, f_a)} \leq \eta_{\text{bute}} \tag{5}\]
where $\eta_{\text{CFAR}}$ is the CFAR threshold, whose computation is presented in Section II-C. From (5), it is pointed out the following [16]:

1) The clutter cancelation is carried out by multiplying each Doppler frequency bin $f_a$ of the signal vector $z(r_k, f_a)$ with the inverse of the CCM $\hat{R}^{-1}_W(f_a)$. The sum of the receive channels before clutter cancelation is given by

$$z_{\text{BCC}}(r_k, f_a) = |d^H(u_t, f_a)z(r_k, f_a)|^2. \quad (6)$$

2) The target matched filtering is performed by multiplying the intermediate result with the Hermitian of the DOA vector $d^H(u_t, f_a)$ (i.e., the expected moving target signal). The sum of the receive channels after clutter suppresion is given by

$$z_{\text{ACC}}(r_k, f_a) = |d^H(u_t, f_a)\hat{R}^{-1}_W(f_a)z(r_k, f_a)|^2. \quad (7)$$

3) After the matched filtering, the Doppler frequency $f_a$, the DOA-angle $\Psi_{\text{DOA, az}}$ and the slant-range $R$ of the moving target signal are estimated.

Exemplarily, Fig. 2 shows the sum of the receive channels before and after the clutter cancelation (i.e., using (6) and (7), respectively). The data patch had the size of one CPI (2048 x 128 range-Doppler samples), and the beamforming vector $d$ was steered to the broadside direction of the array ($u_t = 0$). In this example, the data were centered at zero-Doppler.

**B. PD STAP With A Priori Road Map Information**

Our modified PD STAP [9] processor uses two freely available databases: the OpenStreetMap (OSM) [17] and the digital elevation model (DEM) obtained from the Shuttle Radar Topography Mission [18]. The latter is necessary because the OSM road database does not provide geographical height information.

The OSM provides the geographical positions of the road points, from where it is possible to compute the corresponding road angles ($\alpha_r$) with respect to the UTM Easting axis (cf. Fig. 3). The road point angle allows the computation of the absolute heading velocity of the target (i.e., the velocity of the target on the road), according to [19], [20]

$$v_{\text{head}} = \frac{u_r}{\sin (\theta) \sin (\alpha_r - \alpha_p)} = |v_{\text{abs}}| \quad (8)$$

where $\alpha_p$ is the heading angle of the platform with respect to the UTM easting axis. The moving direction of the target is given by [19], [20]

$$\alpha_t = \begin{cases} \alpha_r, & \text{sgn}(v_{\text{abs}}) = +1 \\ \alpha_r - 180^\circ, & \text{sgn}(v_{\text{abs}}) = -1 \end{cases} \quad (9)$$

where $\text{sgn}(-)$ denotes the sign function.

The decision regarding whether the detections are true (i.e., cars moving on the roads) or not is made based on two positioning error models: one for the PD STAP detections [21] and one for the OSM road points. In this sense, two error ellipses are obtained, as shown in Fig. 3. In Fig. 3, $R_g$ is the slant range of the target projected on the ground and $\theta_{\text{DOA, az}}$ is the DOA-angle of the target projected on the ground and measured with respect to the azimuth direction.

The error models and the equations for computing the axes of both positioning error ellipses are presented in [22]. If both ellipses overlap, the target is considered true and is relocated to the road (i.e., to the closest road point). If the ellipses do not overlap, then the target is discarded as a false detection.

**C. Heterogeneous Clutter Model**

An appropriate and accurate clutter model needs to be fit to the test statistics $\overline{T}$ in order to determine a reliable CFAR detection threshold. Conventional, this fit is done only for the broadside direction of the array (i.e., the beam is steered to $\Psi_{\text{DOA, az}} = 90^\circ$), or, in the case of high squint angles, to the DOA angle corresponding to the clutter Doppler centroid.
For estimating the CFAR threshold, it is important that the test statistics used as "training data" are free of moving target signals and strong discrete scatterers, which may be present due to imperfect clutter cancelation.

This paper uses the heterogeneous clutter model introduced in [23] and further investigated in [24] and [25]. The advantage of this model is that it covers different types of heterogeneity, including completely homogeneous clutter. Hence, it is very flexible for GMTI over land.

In this heterogeneous model, the effective number of looks \( n = E(T) \) and the texture parameter \( \nu \) needs to be estimated from the training data. This texture parameter describes the degree of heterogeneity of the underlying terrain, i.e., the larger the texture parameter \( \nu \), the more homogeneous is the clutter. (Therefore, low values indicate strong heterogeneity.)

The probability density function (PDF) of the adopted heterogeneous model follows a scaled \( F \)-distribution with \( 2n, 2\nu \) degrees of freedom that is given by [23]–[25]

\[
f_X(x) = \frac{\Gamma(n + \nu)}{\Gamma(n)\Gamma(\nu)} \left( \frac{n}{\nu - 1} \right)^n \frac{x^{n-1}}{\left( 1 + \frac{n}{\nu - 1} x \right)^{n+\nu}} \tag{10}\]

where \( \Gamma(\cdot) \) is the gamma function. The texture parameter is estimated according to

\[
\nu = \frac{2nm_2 - (n + 1)}{nm_2 - (n + 1)} \tag{11}\]

where \( m_2 \) is the estimated quadratic mean over all available cells. It is pointed out that for \( \nu \geq 20 \), the \( F \)-distributed heterogeneous model expressed in (10) converges to the \( \chi^2 \)-distributed homogeneous model presented in [23]–[25].

Finally, the CFAR threshold is given by

\[
\eta_{\text{hete}} = \frac{x_0(\nu - 1)}{\nu} \tag{12}\]

where \( x_0 \) is the abscissa value of the cumulative distribution function of (10) at the ordinate \( 1 - P_{\text{fa}} \) and \( P_{\text{fa}} \) is the desired probability of false alarm.

Exemplarily, Fig. 4 shows the histogram of all samples contained in one CPI (2048 \( \times \) 128 range-Doppler samples) along with the scaled \( F \)-distribution given in (10). The estimated parameters were: \( n = 1 \) (which indicates that no multilooking was applied) and \( \nu = 3.61 \) (which suggests a moderately heterogeneous terrain). As it can be seen, the PDF fit very well the measured data, showing that the clutter model was appropriate and accurate.

### III. REAL-WORLD EFFECTS

Training data selection and STAP processing are more challenging when the aircraft is equipped with a low-cost flat antenna array which does not allow electronic or gimbal-based zero-Doppler beam steering. In this case, the time-varying acquisition geometry has to be considered during processing in order to obtain accurate detection, position, and velocity estimates.

In this section, some examples of real-world effects are shown and discussed based on real multichannel X-band data acquired by the DLR’s F-SAR during a GMTI flight campaign conducted in February 2007 [10], [11]. The radar and the geometry parameters are given in Table I (see Section V-A). The SAR image (area on ground: \( 9.4 \times 1.8 \text{ km}^2 \)) is shown in Fig. 5(a).

#### A. Motion of the Aircraft

In reality, the aircraft is not able to follow exactly a straight flight path. Atmospheric turbulences result in time-varying changes of the yaw, pitch, and roll angles, as depicted in Fig. 5(b). Fig. 5(b) shows the angles obtained from the inertial measurement system of the F-SAR during the flight campaign.

Especially the pitch and the yaw angles cause the so-called squint angle \( \Psi_{\text{SQ}} \). The squint angle is both range and time dependent and can be approximately expressed as [26]

\[
\Psi_{\text{SQ}}[r_k, t] \approx \sin^{-1} \left[ \cos(\theta_{[r_k]} + \theta_{\text{ROLL}}[t]) \tan(\theta_{\text{PITCH}}[t]) \right] + \sin(\theta_{[r_k]} + \theta_{\text{ROLL}}[t]) \tan(\theta_{\text{YAW}}[t]) \tag{13}\]

where \( \theta_{[r_k]} = \cos^{-1}(h/R(k)) \) is the incidence angle, \( h \) is the altitude of the aircraft above ground, \( \theta_{\text{YAW}}(t) \), \( \theta_{\text{PITCH}}(t) \), and

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Symbol</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed of light</td>
<td>( c )</td>
<td>( 2.9979 \times 10^3 \text{ m/s} )</td>
</tr>
<tr>
<td>Platform velocity</td>
<td>( v_p )</td>
<td>90 m/s</td>
</tr>
<tr>
<td>Number of receive channels</td>
<td>( M )</td>
<td>4</td>
</tr>
<tr>
<td>Pulse repetition frequency</td>
<td>( \text{PRF} )</td>
<td>2500 Hz</td>
</tr>
<tr>
<td>Range bandwidth</td>
<td>( f_r )</td>
<td>100 MHz</td>
</tr>
<tr>
<td>Effective along-track baseline</td>
<td>( d_a )</td>
<td>0.1 m</td>
</tr>
<tr>
<td>Incidence angle range</td>
<td>( \theta_i )</td>
<td>25° to 63°</td>
</tr>
<tr>
<td>Wavelength</td>
<td>( \lambda )</td>
<td>0.03125 m</td>
</tr>
<tr>
<td>Transmit antenna length</td>
<td>( l_a )</td>
<td>0.5 m</td>
</tr>
<tr>
<td>Altitude of the platform</td>
<td>( h )</td>
<td>2200 m</td>
</tr>
</tbody>
</table>
In reality, also the velocity of the platform $v_p$ may change slightly over time. Thus, during the successive processing of the CPIs, the velocity needs to be updated regularly if no first-order motion compensation [28] is carried out before STAP processing.

Fig. 5(c) shows the change of the Doppler centroid over range and azimuth. The Doppler centroid was estimated using the energy balancing method proposed in [29]. A moving window with $512 \times 2048$ samples with pixel spacing of $3.6$ and $1.2$ m in azimuth and slant-range directions (respectively) was used for the estimation.

In the absence of moving targets in the measured data, comparable results would be obtained between the Doppler centroid estimated from the data and the Doppler centroid obtained from (15). However, it has to be mentioned that the measured data contain vehicles moving on the roads and they were not removed from the data before the Doppler centroid estimation. As a result, a bias is expected in Fig. 5(c) especially along the road axes due to the Doppler shift caused by the moving vehicles. Since Fig. 5(c) has only a demonstrative character for pointing out the need for a range-dependent Doppler centroid correction during processing [27], this issue does not play an important role.

B. Clutter Statistics

Fig. 5(d) shows the change of the clutter statistics according to the texture parameter $\nu$, estimated using (11). Indeed, it can be seen that the lowest textures (i.e., the most heterogeneous patches) were obtained in regions of forests and cities. In contrast, the highest textures (i.e., the most homogeneous patches) were obtained in regions of crop fields, as well as over the lake [see Fig. 5(a)].

Fig. 5(e) shows the threshold $\eta_{hete}$ obtained for $P_{fa} = 10^{-6}$, estimated using (12). Indeed, it can be noticed from Fig. 5(d) and (e) that low texture values cause high CFAR detection thresholds, as verified in [23]–[25].

The results shown in Fig. 5(d) and (e) were obtained by applying a moving window with $128 \times 501$ samples with pixel spacing of $3.6$ and $1.2$ m in azimuth and slant-range directions, respectively. Once more, it is pointed out that the moving target signals were not removed from the moving window, so that a bias may be present especially along the road axes.

Fig. 5(c)–(e) shows that not only the Doppler centroid but also the clutter statistics change over range and azimuth. Thus, additionally to the Doppler centroid correction, the training data must be updated periodically over range and azimuth. This is especially important for achieving a high performance with any STAP algorithm.

C. Channel Imbalances and Along-Track Baselines

In reality, it is not possible to build absolutely identical antennas and receive channels with the same electrical characteristics and time delays. Thus, the transfer functions of the receive channels differ from each other and therefore they need to be characterized or calibrated.

For instance, different transmit and receive antenna characteristics $D_t$ and $D_{r,M}$ can be measured or estimated, and

$$\theta_{ROLL}(t)$$ are the yaw, pitch, and roll angles of the antenna array, respectively. The squint angle causes a range-dependent Doppler shift of the data. Especially for large squint angles, this range dependence can be clearly recognized in the range-Doppler domain as a J-shaped structure, known in the literature as “J-hook” [27].

The relation between the squint angle and the clutter Doppler centroid is given by

$$f_{dc} = \frac{2v_p}{\lambda} \sin(\Psi_{SQ}). \quad (14)$$

Thus, the impacts of aircraft’s attitude angles on the Doppler centroid can be verified by applying (13) into (14)

$$f_{dc,ATT}[r_k, r] \approx \frac{2v_p}{\lambda} \left[ \cos(\theta_i[r_k] + \theta_{ROLL}[r]) \tan(\theta_{PITCH}[r]) + \sin(\theta_i[r_k] + \theta_{ROLL}[r]) \tan(\theta_{YAW}[r]) \right]. \quad (15)$$

Fig. 5. Examples of real-world effects on DLR’s F-SAR data acquisition. (a) SAR image. (b) Change of the yaw, pitch, and roll angles over time. (c) Change of the average Doppler centroid over time and slant range. (d) Change of the texture over time and slant range. (e) Change of the CFAR threshold over time and slant range.
incorporated directly into the beamforming vector $d$. Generally, external calibration is required for compensating different time delays between the channels (e.g., by using reference targets or considering special calibration loops in the radar hardware design). Remaining along-track interferometric (ATI) phase offsets can be estimated afterward from the data.

Furthermore, the precise knowledge of the ATI baselines between the multiple receive channels is important for obtaining accurate DOA-angle estimates.

An elegant method for digital channel balancing was introduced in [30] and discussed in more detail in [31]. In this case, the channels are balanced with respect to a reference channel using an iterative approach in the 2-D frequency domain. Thus, the ATI phase offsets and the amplitude differences are eliminated. In [31], it is shown how the ATI baselines between the individual channels can be estimated accurately in the range-Doppler domain.

For the results presented in Section V, a much simpler and faster (yet not as much accurate) calibration method was used. In this case, only the average amplitude and the ATI phase offset between each pair of channels were estimated. The correction was carried out simply by: 1) normalizing the amplitudes with respect to the amplitude of the reference channel, and 2) multiplying the multichannel data with the corresponding complex conjugated ATI phase offsets.

The amplitude offsets with respect to the first receive antenna are given by

$$\rho_{1,m} = \frac{\max[D_t D_r]_1}{\max[D_t D_r]_m}, \quad m = 2, \ldots, M. \quad (16)$$

In other words, (16) is the ratio between a pair of antenna pattern peaks (taking the first receive antenna as reference).

The results shown in Figs. 6 and 7 were obtained by processing a data patch ($2048 \times 16384$ range–azimuth samples) taken from the data set shown in Fig. 5(a), where a cropping field area was considered (i.e., without moving target signals).

**IV. Training Data Selection and Update**

In Section III, some of the main problems that may arise with airborne data acquisitions were shown and discussed.

**Fig. 6.** Average antenna patterns estimated from real multichannel data. (a) Before and (b) after amplitude correction.

**Fig. 7.** Histograms of the ATI phases between channels 1/2 estimated from real multichannel data: before and after the correction of the ATI phase offset ($\Delta \varphi_{1,2} = -137.8^\circ$). Exemplarily, Fig. 6 shows the average antenna patterns estimated from real multichannel data before and after the correction of the amplitude offsets. In this case, the following amplitude offsets were obtained: $\rho_{1,2} = 1.08$, $\rho_{1,3} = 1.01$, and $\rho_{1,4} = 1.05$. The correction of the amplitude offsets is important since the amplitudes of the antenna patterns ($D_t$ and $D_r$) are contained in the beamforming vector $d(1)$.

For example, Fig. 7 shows the histograms of the ATI phases between channels 1/2 before and after the correction of the ATI phase offset. As it can be seen, after correcting the phase offset the histogram is centered at zero (i.e., the mean ATI phase is zero). In this case, the following ATI phase offsets were obtained: $\bar{\Delta \varphi}_{1,2} = -137.82^\circ$, $\bar{\Delta \varphi}_{1,3} = 14.26^\circ$, and $\bar{\Delta \varphi}_{1,4} = 70.16^\circ$.

The results shown in Figs. 6 and 7 were obtained by processing a data patch ($2048 \times 16384$ range–azimuth samples) taken from the data set shown in Fig. 5(a), where a cropping field area was considered (i.e., without moving target signals).
Indeed, the clutter statistics change over range and azimuth is a challenging concern since it impacts the STAP performance. Thus, the training data must be updated regularly taking into account potential changes in the clutter statistics.

This section presents four algorithms (or strategies) for selecting the training data for CCM estimation. The algorithms 1 and 2 are examples of conventional methods employed for STAP, while the algorithms 3 and 4 are novels. Section IV-E presents a module for moving target signal rejection (MTSR) that can be applied to all algorithms.

A. Algorithm 1 (Conventional)

In this algorithm, the CCM is estimated once per CPI using all the samples of the CPI as training data, as shown in Fig. 8. The main steps for processing one CPI are listed in the following.

1) Obtain the CPI by partitioning the time domain multichannel data in the azimuth direction (e.g., \( n_a \) = 128 azimuth samples). Use all the available range bins \( n_r \).
2) Obtain the training data by using the full content of the CPI. Next, transform the training data to range-Doppler domain via an azimuth FFT.
3) Estimate the CCM according to (3), using \( K = n_r \).
4) Apply the PD STAP processor on the CPI and estimate the CFAR threshold (as described in Section II-C).
5) Detect the moving targets using the CFAR threshold from Step 4. Finally, estimate their parameters (as described in Section II-A).

It is pointed out that this algorithm does not perform data selection at all, since the full CPI is used as training data. However, the MTSR module (see Section IV-E) can be applied in order to reject the range bins that contain moving targets or strong scatterers from the training data.

B. Algorithm 2 (Conventional)

The principle of this algorithm is shown in Fig. 9. As it can be seen, this algorithm applies a moving window along range for each CPI, so that the training data (red boxes in Fig. 9) are updated for each cell under test (CUT). In this sense, for each CUT, \( K_g \) range bins are used as guard zones and \( K_2 \) range bins are selected as training data (e.g., \( K_g = 2 \) and \( K_2 = 128 \) range bins). The main steps for processing one CUT are listed in the following.

1) Obtain the CUT by selecting one range bin of the CPI.
2) Obtain the training data by selecting \( K_2 \) range bins around the CUT and the guard zones. Next, transform the training data to range-Doppler domain via an azimuth FFT.
3) Estimate the CCM according to (3), using \( K = K_2 \).
4) Apply the PD STAP processor on the training data just for estimating the CFAR threshold.
5) Apply the PD STAP processor on the CUT and detect the moving targets using the CFAR threshold from Step 4. Finally, estimate the parameters of the moving targets.

This algorithm takes into account the clutter change of the scene, since the CCM (and therefore the CFAR threshold) is adaptively updated for each CUT as the moving window slides over range. It is pointed out that this algorithm requires the highest computational effort among all algorithms presented in this paper.

Furthermore, a tradeoff is observed regarding the chosen number of range bins \( K_2 \) of the training data. For large \( K_2 \), the algorithm loses robustness against the clutter change. For small \( K_2 \), the training data may be insufficient for estimating reliable CFAR thresholds. Special attention is required if the MTSR module is applied, since it decreases even further the number of range bins of the training data.

An alternative to overcome this issue is presented by the proposed algorithms 3 and 4, where several CPIs are used as training data inside a moving window.

C. Algorithm 3 (Novel)

The principle of this algorithm is shown in Fig. 10. This algorithm applies a moving window in the azimuth direction,
where the length $L = 2T + D$ of the window determines the number of CPIs used as training data (red box in Fig. 10). The CPIs marked as $D$ (in blue) are processed by the PD STAP, while the adjacent CPIs marked as $T$ (in red) are used only as training data. In order to reduce the computational effort, the moving window slides after all CPIs $D$ are processed by the PD STAP. Note that all the available range bins $n_r$ are used.

The training data array is built by: 1) “stacking” the $L$ CPIs in time domain and 2) transforming the data array to range-Doppler domain via an azimuth FFT (see detail in Fig. 10). The goal of this technique is to increase the number of range bins of the training data by a factor of $L$ (i.e., the training data have $n_rL \times f_a, \text{CPI}$ range-Doppler samples). The main steps of this algorithm are listed in the following.

1) Select the data patch with $L$ CPIs by partitioning the multichannel data in time domain, in the azimuth direction.

2) Build the training data array by “stacking” the $L$ CPIs.

3) Estimate the CCM according to (3), using $K = K_3 = n_rL$.

4) Apply the PD STAP processor on the training data and estimate the CFAR threshold.

5) Apply the PD STAP processor on the CPIs $D$ and detect the moving targets using the CFAR threshold from Step 4. Finally, estimate the parameters of the moving targets.

The use of the MTSR module is very promising in this case, since the “cleaned” training data still contain enough samples for estimating reliable CFAR thresholds. However, this algorithm does not take into account the clutter change over range, since all the available range bins $n_r$ are used. For this reason, algorithm 4 is proposed as an alternative to solve this issue.

**D. Algorithm 4 (Novel)**

This algorithm works similarly as algorithm 3, whereas it presents the flexibility to change the number of range bins of the moving window for taking into account the clutter change over range.

The principle of this algorithm is shown in Fig. 11. As it can be seen, the data patches are processed as the moving window slides over range and azimuth. It is pointed out that the data patches are partitioned without overlap in range in order to decrease the computational effort.

In this case, the training data (red box in Fig. 11) contain $L = 2T + D$ truncated CPIs, where only $n'_r$ range bins are used. The training data array is built by “stacking” the $L$ CPIs, as shown in Fig. 11 (detail). The CCM is estimated according to (3), using $K = K_4 = n'_rL$. The use of the MTSR module is also very promising in this case.

It is important to mention that the length of the moving window (i.e., the choice of the parameters $D$, $T$, and $n'_r$) plays a big role for the CCM estimation. In this sense, the impacts caused by different moving window lengths are shown and discussed in Section V-B using real multichannel data. The comparison of all algorithms is presented in Section V-E.

**E. Moving Target Signal Rejection**

The MTSR is an optional module that can be applied on all previously described algorithms. The objective is to remove undesired range bins of the training data that contain moving target signals and strong scatterers, so that cleaner training data can be obtained. The principle of the MTSR module is similar to the GIP test presented in [8], including a comparable increase of the processing time.

Fig. 12 (left) shows a simplified workflow of the proposed MTSR module, where four main steps are carried out.

**Step 1:** The PD STAP processor is applied on the original uncleaned training data in order to detect the range bins of the moving targets.

**Step 2:** The magnitudes of all samples contained in the uncleaned training data are sorted (e.g., in descending order).

**Step 3:** The number of undesired range bins to be rejected from the uncleaned training data is established by an empirical threshold $\xi$, which is manually set (e.g., $\xi = 205$ undesired range bins).

![Fig. 11. Principle of algorithm 4 (novel). The training data (red box) are updated as the window slides over range and azimuth. The truncated CPIs $D$ are processed by the PD STAP.](image)

![Fig. 12. (Left) Workflow of the proposed MTSR module. (Right) Application example based on measured data containing strong scatterers and moving target signals. This example shows (a) uncleaned training data in range-Doppler domain and (b) cleaner training data obtained after applying the MTSR module. (Note that 205 undesired range bins were rejected.) The cleaner training data are used for the CCM estimation.](image)
Step 4: The undesired range bins containing moving target signals (obtained from Step 1) are first rejected. Then, the range bins containing the strongest scatterers (sorted in Step 2) are rejected until the threshold $\xi$ is reached.

The experimental results presented in Section V show that the empirical threshold $\xi$ worked very well for different data sets with different numbers of moving targets.

Exemplarily, Fig. 12(a) shows an application example where the uncleaned training data (with 2048 × 128 range-Doppler samples) contained moving target signals and strong scatterers (e.g., corner reflectors—CRs). In this example, the empirical threshold was set to $\xi = 205$ undesired range bins (i.e., $\approx 10\%$ of the total number of available range bins). Therefore, after applying the MTSR module, cleaner training data were obtained with 1843 azimuth lines, as shown in Fig. 12(b). In Fig. 12(b), note that the moving target signals and most of the strong scatterers were rejected (including the CRs). The cleaner training data are then used for the CCM estimation.

An analysis of the impact of the MTSR module on the clutter model using real multichannel data is presented in Section V-C.

V. EXPERIMENTAL RESULTS

In this section, the training data selection algorithms are compared using three data sets (real multichannel X-band data) that contain five controlled cars with different positions and velocities. An analysis of the moving window parameters is presented for algorithms 3 and 4, and the impact of the MTSR module on the estimated texture and CFAR threshold is evaluated. GMTI detection results are presented in order to highlight (visually) the main differences among the algorithms and a quantitative comparison is carried out taking into account the number of true and false detections, and the estimated probability of false alarm. Finally, GMTI results are shown for a real traffic scenario on a highway with dozens of vehicles of opportunity.

A. Experimental Setup

The flight campaign of DLR’s airborne F-SAR was conducted over the Allgäu airport in Memmingen in February 2007. Fig. 13 shows an optical image of the airport’s runway (left), where five controlled cars were moving. On the right, the positions and the velocities of the cars are shown for each data set, where the angle of the runway with respect to the flight path was also different. Note that car 5 moved off-road (in circles) in the data takes 1 and 2.

It has to be mentioned that cars 1, 2, and 4 were equipped with radar reflectors in order to enhance their RCS, and car 4 had a differential GPS receiver for retrieving reliable geographical reference positions and velocities. A detailed experiment description is found in [19] and [32]. The numbers of the processed range and azimuth samples ($N_r$ and $N_a$, respectively) of the data sets are as follows.

1) Data Set 1: $N_r = 1024$, $N_a = 16384$
2) Data Set 2: $N_r = 1024$, $N_a = 32768$
3) Data Set 3: $N_r = 1024$, $N_a = 32768$

The radar parameters are given in Table I. The effective PRF per RX channel was 2.5 kHz. The data sets were processed using CPIs of 1024 × 128 range–azimuth samples, and the beamformers were applied using DOA angle steps of 0.05° within an interval determined by the azimuth antenna beamwidth. The probability of false alarm of the CFAR detector was set to $P_{fa} = 10^{-6}$.

B. Parameters of the Moving Window

The choice of the moving window parameters ($D$, $T$, and $n'_r$) for algorithms 3 and 4 plays a big role since it defines not only the amount of training data used for the CCM estimation, but also how often the training data are updated. In this section, the data set 1 was processed by algorithms 3 and 4 (with MTSR), where different moving window lengths were applied. The PD STAP processor with road map information was used in the following experiments for counting the number of detections relocated to the
the number of range bins \( n \) (i.e., True #1), the data set 1 was repeatedly processed by algorithm 3 (with MTSR) assuming \( D = 15 \) and \( T = 12 \). The positions of the cars were known. The detections of car 5 were not counted since it moved off-road and thus its detections were discarded after applying the road map information.

Fig. 14(a) shows the number of true detections of cars 1–4 (i.e., True\(_{\bar{1} \rightarrow 4}\)) as a function of the parameters \( D \) and \( T \). These detections could be counted as true since the velocities and the positions of the cars were known. The detections of car 5 were not counted since it moved off-road and thus its detections were discarded after applying the road map information.

Fig. 14(b) shows the number of false detections relocated to the runway as a function of the parameters \( D \) and \( T \). In this case, the numbers of false detections were obtained by subtracting the number of true detections (True\(_{\bar{1} \rightarrow 4}\)) from the number of total detections.

Fig. 14 shows that both the number of true and false detections increases proportionally with the length of the moving window. For instance, a good compromise can be achieved when \( D = 15 \) and \( T = 12 \), where 450 true detections and only two false detections were obtained. Therefore, these parameters were chosen for the moving window applied on algorithms 3 and 4, so that \( L = 2T + D = 39 \) CPIs (i.e., nearly two seconds of data).

2) Experiment 2 (Number of Range Bins): In this experiment, the data set 1 was repeatedly processed by algorithm 4 (with MTSR) assuming \( D = 15 \) and \( T = 12 \), and varying the number of range bins \( n' \) of the moving window. Once more, only the detections from cars 1–4 were counted as true (True\(_{\bar{1} \rightarrow 4}\)).

Fig. 15(a) shows that the number of detections (all, relocated, and true) obtained from each moving window. As it can be seen, the levels increased as the number of range bins \( n' \) increased, reaching the peak at \( n'_r = 512 \). In other words, up to this point \((n'_r \leq 512)\), the CCM estimation benefited from the increase of training data. Beyond this point \((n'_r > 512)\), algorithm 4 lost its robustness against the clutter change over range. As a result, the number of detections (all, relocated and true) started decreasing. Note that the numbers of range bins \( n'_r \) were applied as a power of two, since the window moved without overlap over range.

The same trend is observed in the percentages of detections shown in Fig. 15(b) [calculated using the values from Fig. 15(a)]. Moreover, the results obtained with data sets 2 and 3 followed the same trend as for data set 1 (whose results are shown in Figs. 14 and 15).

The chosen parameter for algorithm 4 (used for obtaining the results presented in Sections V-D–V-F) was \( n'_r = 512 \).

The results obtained from experiments 1 and 2 highlighted the impacts of the moving window length on real multichannel data. However, in reality, the most suitable moving window length depends on the scene and on the motion of the aircraft. Therefore, the parameters chosen in this subsection \((D = 15, T = 12, \) and \( n'_r = 512)\) were matched to data set 1.

C. Impacts of MTSR on Clutter Model

In this section, the conventional algorithm 1 was used to process the data set 1. As described in Section IV-A, this algorithm estimates the CCM (and therefore the texture and the CFAR threshold) once per CPI. In other words, by using this algorithm, it is possible to verify the texture and the CFAR threshold change over time (1 CPI \( \approx 0.05 \) s of data).

Fig. 16(a) shows the texture change over time [estimated using (11)] with and without applying the MTSR module. As expected, higher texture values (i.e., more homogeneous) were obtained after applying the MTSR module, since moving target signals and strong scatterers were removed from the training data. In addition, the texture change over time was smoother when the MTSR module was applied.

Fig. 16(b) shows the CFAR threshold change over time [estimated using (12)] with and without applying the MTSR module. As pointed out in Section II-B, it is possible to note that the lower the texture (i.e., the more heterogeneous), the higher the CFAR threshold. In this sense, the decrease of the CFAR threshold after applying the MTSR module was expected, since the textures were higher. Moreover, the CFAR threshold change was much smoother after applying the MTSR module.

In this experiment, 205 undesired range bins were rejected from the training data (i.e., \( \xi = 205 \)).
Fig. 17. Google Earth images overlaid with PD STAP detections. (a)–(d) Algorithms 1–4 (without MTSR) were applied for selecting the training data for the CCM estimation. The detail shows the (e) relocation of the detections (circles) to their closest OSM road points (triangles) [9], [22]. The center of the runway (white line) was the OSM road axis.
For data set 1, it is pointed out that the runway with cars 1–4 was located in the center of the image (i.e., around 3.3 s of data). Therefore, around this time instant, it is possible to notice in Fig. 16 that the texture reached its minimum (i.e., most heterogeneous point) and the CFAR threshold reached its maximum.

The MTSR module has the benefit to make the texture and the CFAR threshold less sensitive to moving target signals and strong scatterers. In this sense, these parameters could be estimated less often in order to save processing time. However, the MTSR module itself increases the processing time, especially because the PD STAP processor needs to be applied on the training data only for detecting the range bins containing the moving targets.

D. GMTI Results for Data Set 1 (Runway at 90°)

The PD STAP detections obtained by algorithms 1–4 are shown in Fig. 17. In Fig. 17, the colors of the detections are related to their absolute ground range velocities. In Fig. 17(e), a detail is shown for algorithm 4 in order to demonstrate the relocation of the targets to the road axis of the OSM, carried out by the PD STAP processor with road map information [9]. In this case, the radar detections are shown before (circles) and after (triangles) relocation using the OSM database, where the center of the runway (white line) was considered as road axis. The triangles point to the heading angles of the cars and the thin yellow lines connect the PD STAP detections to their closest road points (i.e., on the road axis).

Fig. 18 shows the eigenvalues (normalized to the noise power) obtained from the training data selected by the four algorithms for the CCM estimation. In this case, the processed data patch contained part of the runway where the cars 1–4 moved, so that the impact of the moving target signals can be verified in the eigenvalue distributions. For algorithms 1 and 2, the eigenvalue distributions exhibited spikes due to the influence of moving target signals and strong scatterers. The highly contaminated eigenvalue distributions obtained for algorithm 2 explain why this algorithm presented the highest number of false detections [see Fig. 17(b)]. In contrast, for algorithms 3 and 4, the eigenvalue distributions were very smooth due to the large amount of training data used by these algorithms.

The eigenvalue distributions can also be useful for verifying the clutter suppression capability of the PD STAP processor, which in this case is given by the highest difference between the first and the second eigenvalues (≈15 dB). The clutter suppression capability could be further improved, for instance, by using more sophisticated digital channel calibration techniques [31]. Nevertheless, this is out of scope for this paper. The fourth eigenvalue is in the noise power level.

The benefits of the MTSR module can be verified through the ground range velocity histograms of the PD STAP detections, shown in Fig. 19. Fig. 19 also allows comparing the amount of false (blue bars) and true detections (red bars) obtained for all algorithms. Once more, the true detections (i.e., True#1−5) could be counted since the positions and the velocities of all cars were known a priori.

From Fig. 19, it can be seen the following.

1) Algorithm 1 benefited from the MTSR, mainly because car 3 could be detected (i.e., its self-whitening was solved after selecting the training data with the MTSR module). Nevertheless, note that in this case the number of true and false detections increased.

2) Algorithm 2 did not benefit from the MTSR, since its amount of training data was further decreased. As a result, car 1 was barely detected and car 3 could not
Fig. 19. Histograms of the ground range velocities of the detections obtained for all algorithms (data set 1): all (blue bars) and true (red bars) detections. The detections obtained from cars 1–5 are numbered in the histograms.

3) Algorithms 3 and 4 benefited from the MTSR and presented the best results. The MTSR module increased the number of true detections (especially from the slower cars 1 and 3), whereas the number of false detections also increased.

In general, all algorithms presented accurate velocity estimates of all cars. The estimated positions were also accurate considering that the cars 1–4 moved on the edges of the runway (see Fig. 13, data set 1).

This section showed that the MTSR module increased the number of true and false detections for algorithms 1, 3, and 4. In contrast, the number of true and false detections for algorithm 2 decreased. A quantitative comparison among all algorithms is presented in Section V-E in order to clarify in numbers the pros and cons of the MTSR module.

**E. Comparison of All Algorithms**

A quantitative comparison among all algorithms is shown in Tables II–IV, which summarize for all data sets:

1) the number of all detections obtained by the PD STAP processor;
2) the number of relocated detections to the runway;
3) the number of true detections;
4) the estimated probability of false alarm $\hat{P}_{fa}$, which is shown with respect to the desired probability of false alarm of the CFAR detector (i.e., $P_{fa} = 10^{-6}$);

### TABLE II

**NUMBER OF DETECTIONS FROM DATA SET 1 (RUNWAY AT 90°)**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>All</th>
<th>Rel.</th>
<th>True (#1-4)</th>
<th>True (#1-5)</th>
<th>$\hat{P}_{fa}$ [x10^6]</th>
<th>True_{all}/Rel. [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>134</td>
<td>53</td>
<td>51</td>
<td>77</td>
<td>3.40</td>
<td>96.23</td>
</tr>
<tr>
<td>2</td>
<td>2470</td>
<td>1195</td>
<td>1073</td>
<td>1208</td>
<td>75.22</td>
<td>89.79</td>
</tr>
<tr>
<td>3</td>
<td>223</td>
<td>181</td>
<td>180</td>
<td>203</td>
<td>1.19</td>
<td>99.45</td>
</tr>
<tr>
<td>4</td>
<td>188</td>
<td>151</td>
<td>149</td>
<td>171</td>
<td>1.01</td>
<td>98.68</td>
</tr>
<tr>
<td>1 (MTSR)</td>
<td>880</td>
<td>539</td>
<td>522</td>
<td>593</td>
<td>17.11</td>
<td>96.85</td>
</tr>
<tr>
<td>2 (MTSR)</td>
<td>1439</td>
<td>834</td>
<td>760</td>
<td>800</td>
<td>38.09</td>
<td>91.13</td>
</tr>
<tr>
<td>3 (MTSR)</td>
<td>573</td>
<td>450</td>
<td>448</td>
<td>487</td>
<td>5.13</td>
<td>99.55</td>
</tr>
<tr>
<td>4 (MTSR)</td>
<td>625</td>
<td>507</td>
<td>502</td>
<td>537</td>
<td>5.25</td>
<td>99.01</td>
</tr>
</tbody>
</table>

1) be detected anymore. In this case, note that the MTSR module decreased both the number of true and false detections.

### TABLE III

**NUMBER OF DETECTIONS FROM DATA SET 2 (RUNWAY AT 45°)**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>All</th>
<th>Rel.</th>
<th>True (#1-4)</th>
<th>True (#1-5)</th>
<th>$\hat{P}_{fa}$ [x10^6]</th>
<th>True_{all}/Rel. [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>457</td>
<td>100</td>
<td>86</td>
<td>101</td>
<td>10.61</td>
<td>86.00</td>
</tr>
<tr>
<td>2</td>
<td>6897</td>
<td>1819</td>
<td>1574</td>
<td>2150</td>
<td>141.47</td>
<td>86.53</td>
</tr>
<tr>
<td>3</td>
<td>396</td>
<td>188</td>
<td>182</td>
<td>269</td>
<td>3.78</td>
<td>96.81</td>
</tr>
<tr>
<td>4</td>
<td>388</td>
<td>106</td>
<td>105</td>
<td>150</td>
<td>7.09</td>
<td>99.06</td>
</tr>
<tr>
<td>1 (MTSR)</td>
<td>3052</td>
<td>932</td>
<td>806</td>
<td>1423</td>
<td>48.55</td>
<td>86.48</td>
</tr>
<tr>
<td>2 (MTSR)</td>
<td>3905</td>
<td>1153</td>
<td>1006</td>
<td>1434</td>
<td>73.64</td>
<td>87.25</td>
</tr>
<tr>
<td>3 (MTSR)</td>
<td>1102</td>
<td>589</td>
<td>586</td>
<td>900</td>
<td>6.02</td>
<td>99.49</td>
</tr>
<tr>
<td>4 (MTSR)</td>
<td>957</td>
<td>498</td>
<td>497</td>
<td>690</td>
<td>7.96</td>
<td>99.80</td>
</tr>
</tbody>
</table>

### TABLE IV

**NUMBER OF DETECTIONS FROM DATA SET 3 (RUNWAY AT 30°)**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>All</th>
<th>Rel.</th>
<th>True (#1-5)</th>
<th>True (#1-5)</th>
<th>$\hat{P}_{fa}$ [x10^6]</th>
<th>True_{all}/Rel. [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>478</td>
<td>350</td>
<td>243</td>
<td>73</td>
<td>7.60</td>
<td>69.43</td>
</tr>
<tr>
<td>2</td>
<td>3328</td>
<td>1985</td>
<td>1606</td>
<td>51</td>
<td>51.32</td>
<td>80.91</td>
</tr>
<tr>
<td>3</td>
<td>302</td>
<td>224</td>
<td>193</td>
<td>3.25</td>
<td>86.16</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>223</td>
<td>135</td>
<td>115</td>
<td>3.22</td>
<td>85.19</td>
<td></td>
</tr>
<tr>
<td>1 (MTSR)</td>
<td>1869</td>
<td>1270</td>
<td>972</td>
<td>26.73</td>
<td>76.54</td>
<td></td>
</tr>
<tr>
<td>2 (MTSR)</td>
<td>1752</td>
<td>1076</td>
<td>884</td>
<td>25.87</td>
<td>82.16</td>
<td></td>
</tr>
<tr>
<td>3 (MTSR)</td>
<td>431</td>
<td>342</td>
<td>312</td>
<td>3.54</td>
<td>91.23</td>
<td></td>
</tr>
<tr>
<td>4 (MTSR)</td>
<td>282</td>
<td>187</td>
<td>169</td>
<td>3.37</td>
<td>90.38</td>
<td></td>
</tr>
</tbody>
</table>
the percentage of true detections relocated to the runway, which is very useful for comparing all algorithms (with and without applying the MTSR module).

The number of relocated detections to the airport’s runway was counted after applying the PD STAP processor with road map information (see Section II-B). Note that for data sets 1 and 2, only the cars 1–4 were relocated to the runway (since car 5 moved off-road).

The $\hat{P}_{\text{fa}}$ was estimated according to

$$\hat{P}_{\text{fa}} = \frac{\text{All} - \text{True}_{1-5}}{N_{a}N_{r}}. \quad (20)$$

From Tables II–IV, it can be seen the following.

1) The MTSR increased the $\hat{P}_{\text{fa}}$ of algorithms 1, 3, and 4 for all data sets. In other words, it means that the number of false detections increased more than the number of true detections. However, it was shown in Section V-D that the MTSR module applied on algorithm 1 avoided the self-whitening of car 3. Thus, since the $\hat{P}_{\text{fa}}$ obtained for algorithms 3 and 4 after applying the MTSR were tolerable, the MTSR is recommended for algorithms 1, 3, and 4.

2) The MTSR decreased the $\hat{P}_{\text{fa}}$ of algorithm 2 on all data sets by almost half, which at a first glance looks as an improvement. However, several true detections were lost after applying the MTSR module (e.g., car 3 could not be detected), and even so this algorithm presented the worst $\hat{P}_{\text{fa}}$. Thus, the MTSR is not recommended for algorithm 2.

3) The last column shows the percentage of true detections that remained in the final image after applying the road map information. In this case, the MTSR applied on algorithms 3 and 4 presented the best results for all data sets. In the worst case scenario (for data set 3), 90% of the detections were true using algorithm 4 (with MTSR). These values show the great potential of our PD STAP processor with road map information [9], [22].

The amount of training data is of great importance for the CCM estimation. In this sense, it was expected that the proposed algorithms 3 and 4 presented the best $\hat{P}_{\text{fa}}$ and the best percentages of true detections for all data sets. It is pointed out that a threshold was applied in order to reject detections with SCNRs lower than 8 dB, as described in [19].

F. Real Traffic Scenario

In this last experiment, our PD STAP processor with road map information was applied on a data patch (2048 × 16384 range–azimuth samples) taken from the large data set shown in Fig. 5(a), where a part of the highway A7 is contained with several moving cars. Algorithm 4 (with MTSR) was applied for selecting the training data for the CCM estimation and the following moving window parameters were used: $D = 15$, $T = 12$, and $n_{r}' = 512$ (as discussed in Section V-B).

Fig. 20 shows the GMTI results, where the moving cars are depicted as colored triangles pointing to their heading angles and the highway A7 is shown in white. The colors are related to the absolute velocities of the cars on ground $|v_{\text{abs}}|$, estimated according to (8). The information box in Fig. 20 shows some examples of parameters that are obtained for each car on the highway.

For instance, it is possible to verify the traffic situation in both directions of the highway through the ground range velocity histograms of the PD STAP detections, as shown in Fig. 21(a). Fig. 21 shows the histograms of the total (blue bars) and the relocated (red bars) detections on the highway. As it can be seen, the traffic toward Ulm (i.e., positive ground range velocities) was more intense than the traffic toward Memmingen (i.e., negative ground range velocities).
It is also possible to verify the histograms of the signal-to-clutter plus noise ratio (SCNR) of the detections, as shown in Fig. 21(b). Fig. 21(b) shows the histograms of the total (blue bars) and the relocated (red bars) detections on the highway. In both cases, the mean SCNR of the cars was about 27 dB. Once more, it is pointed out that a threshold was applied in order to reject detections with SCNRs lower than 10 dB [20].

No ground truth data were available in this experiment, and therefore it is not possible to determine the percentage of true detections, the probability of false alarm and the errors of the estimated parameters (e.g., velocities and positions). However, the estimated velocities on the highway A7 are reasonable.

Finally, it can be seen from Figs. 17 and 20 that the PD STAP processor was able to detect each single car several times, revealing the potential of this processor for traffic monitoring applications. It has to be pointed out that no clustering or tracking algorithms were applied for obtaining these results.

VI. CONCLUSION

This paper compares four training data selection algorithms for CCM estimation, where two are conventional (used as reference) and two are novels (based on a moving window). Besides, a module for moving target signals rejection (MTSR) is presented for removing the range bins of the training data that contain moving target signals or strong scatterers. The algorithms are applied on PD STAP processors without and with a priori road map information, in both cases using real four-channel data acquired by the DLR’s airborne F-SAR.

It is shown that the two novel algorithms outperform the conventional ones by presenting lower probabilities of false alarm rates and higher percentages of true detections. For the proposed algorithms, the MTSR module increases the number of true detections (avoiding the self-whitening of moving targets) and keeps a tolerable probability of false alarm. Even in the worst case scenario, 90% of the detections that remained in the final image were true. Thus, the main goal of the proposed training data selection and update algorithms is fulfilled.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions that improved both clarity and precision of this paper significantly.

REFERENCES

on “SAR Moving Target Techniques” for the Carl-Cranz-Gesellschaft and since 2016 he is also active at the Friedrich-Alexander-University Erlangen, Erlangen, Germany. He is currently the Lead of a maritime security project, with a focus on the development of novel ship detection and inverse SAR (ISAR) algorithms using airborne radars operated along arbitrary flight tracks. He has authored or co-authored over ten peer-reviewed journal papers, about 55 conference papers and an invited book chapter on multichannel SAR for GMTI in the Academic Press Library in Signal Processing. He holds one patent in the field of SAR-MTI. His research interests include the development of novel MTI algorithms for road and maritime traffic monitoring applications using single- and multichannel air- and spaceborne SARs, along-track interferometry, space–time adaptive processing, ISAR imaging, digital beamforming on receive, time–frequency analysis, and other advanced signal and image processing techniques.

Dr. Baumgartner successfully participated in the DLR Graduate Program, a DLR high-quality qualification initiative that accompanies the doctoral phase. In 2016, he was selected for the DLR Talent Management Program, a high caliber and very individual personnel development tool which provides outstanding employees with intensive support over the course of two years.