Finding Geometric Models by Clustering in the Consensus Space

Daniel Barath\textsuperscript{1}, Denys Rozumnyi\textsuperscript{2,1}, Ivan Eichhardt\textsuperscript{3,4}, Levente Hajder\textsuperscript{3}, Jiri Matas\textsuperscript{2}

\textsuperscript{1}Computer Vision and Geometry Group, ETH Zurich, Switzerland, \textsuperscript{2}VRG, Faculty of Electrical Engineering, CTU in Prague, Czech Republic, \textsuperscript{3}TMEIC Corporation Americas, Roanoke, VA, USA \textsuperscript{4}E"otv"os Lor"and University, Budapest, Hungary

Abstract

We propose a new algorithm for finding an unknown number of geometric models, e.g., homographies. The problem is formalized as finding dominant model instances progressively without forming crisp point-to-model assignments. Dominant instances are found via a RANSAC-like sampling and a consolidation process driven by a model quality function considering previously proposed instances. New ones are found by clustering in the consensus space. This new formulation leads to a simple iterative algorithm with state-of-the-art accuracy while running in real-time on a number of vision problems – at least two orders of magnitude faster than the competitors on two-view motion estimation. Also, we propose a deterministic sampler reflecting the fact that real-world data tend to form spatially coherent structures. The sampler returns connected components in a progressively densified neighborhood-graph. We present a number of applications where the use of multiple geometric models improves accuracy. These include pose estimation from multiple generalized homographies; trajectory estimation of fast-moving objects; and we also propose a way of using multiple homographies in global SfM algorithms. Source code: \url{https://github.com/danini/clustering-in-consensus-space}.

1. Introduction

Robust multi-instance model fitting is the problem of interpreting a set of data points as a mixture of noisy observations stemming from multiple instances of geometric models. Examples for such a problem are the estimation of plane-to-plane correspondences (i.e., homography matrices) in two images, and the retrieval of rigid motions in a dynamic scene captured by a moving camera. In the state-of-the-art algorithms, finding an unknown number of model instances is achieved by clustering the data points into disjoint sets, each representing a particular model instance. Robustness is achieved by considering an outlier model.

Multi-instance model fitting has been studied since the early sixties. The Hough-transform \cite{22,23} is perhaps the first popular method for finding multiple instances of a single class \cite{18,37,44,67}. The RANSAC \cite{16} algorithm was as well extended to deal with finding multiple instances. Sequential RANSAC \cite{25,60} detects instances in a sequential manner by repeatedly running RANSAC to recover a single instance and, then, removing its inliers from the point set. The greedy approach that makes RANSAC a powerful tool for recovering a single instance becomes its drawback when estimating multiple ones. Points are assigned not to the best but to the first instance, typically the one with the largest support, for which they cannot be deemed outliers. MultiRANSAC \cite{71} forms compound hypotheses about $n$ instances. In each iteration, MultiRANSAC draws samples of size $n$ times $m$, where $m$ is the number of points required for estimating a model instance, e.g., $m = 4$ for homographies. Besides requiring the number $n$ of the instances to be known a priori, the increased sample size affects the problem complexity and, thus, the processing time severely.

Modern approaches for multi-model fitting \cite{1,3,24,32–34,40,62,66} follow a two-step procedure. First, they gen-
generate many instances by repeatedly selecting minimal point sets and fitting model instances. Second, a subset of the hypotheses is selected interpreting the input data points the most. This selection is done in various ways. For instance, a popular group of methods \cite{1,3,24,40} optimizes point-to-model assignments by energy minimization using graph labeling techniques \cite{8}. The energy originates from point-to-model residuals, label costs \cite{14}, and geometric priors \cite{40} such as the spatial coherence of the data points. Another group of methods uses preference analysis based on the distribution of the residuals of data points \cite{32–34,68}. Also, there are techniques \cite{62,63,69} approaching the problem as hyper-graph partitioning where the instances are represented by vertices, and the points by hyper-edges.

Prog-X \cite{4} and CONSAC \cite{26} discussed that the first, instance generation, step of the mentioned methods leads to a number of issues, \textit{e.g.}, the instances are generated blindly, having no information about the data at hand. This approach severely restricts the out-of-the-box applicability of such techniques since the user either has to consider the worst-case scenario and, thus, generate an unnecessarily high number of instances; or requires some rule of thumb, \textit{e.g.}, to generate twice the point number hypotheses that provides no guarantees of finding the sought instances. Prog-X approaches the problem via interleaving the model proposal and optimization steps. CONSAC further improves it by using a deep-learning-based guided sampling approach.

A common point of \textit{all} state-of-the-art algorithms is formalizing the multi-model fitting problem as finding disjoint sets of data points each representing a model instance. There are two main practical issues with this assumption. First, in some cases, a point belongs to multiple instances and this assumption renders the problem unsolvable, see the left image of Fig. 2. Also, the point-to-model assignment is often unclear even if it is done by a human, especially, for points around the intersection of instances, see the right two plots of Fig. 2 for examples. The second issue stems from the recovery of disjoint point sets that usually requires a rather complex procedure, \textit{e.g.} labeling via energy minimization, that affects the run-time severely.

The main contribution of this paper is a fundamentally new problem formulation that does not require forming crisp point-to-model assignments, \textit{i.e.}, a point can be assigned to multiple instances. This is different from the formulations used in the state-of-the-art algorithms for general multi-model fitting \cite{1,3,4,24,26,34,40,63}. This property allows the proposed method to be a simple iterative algorithm and, yet, to obtain results superior to the state-of-the-art both in terms of accuracy and run-time, being \textit{real-time} on a number of problems, see Fig. 1, including ones where multi-model fitting algorithms generally are \textit{not} real-time, \textit{e.g.}, two-view motion detection. Also, this assumption relaxes the greedy nature of sequential algorithms as the ordering in which the instances are proposed becomes unimportant. As the second contribution, we discuss ways of exploiting multiple instances in popular applications – Structure-from-Motion, pose estimation for generalized and pin-hole cameras, and trajectory estimation of fast-moving objects. By considering multiple models, the accuracy is increased in almost all cases on several publicly available real-world datasets. As the third contribution, we propose a new sampler designed specifically for multi-instance model fitting. The sampler considers that real-world data tend to form spatially coherent structures. It returns the connected components in a gradually densified neighborhood-graph. While several samplers exist that exploit spatial properties of the data, \textit{e.g.} \cite{6,38}, the proposed one is \textit{deterministic}.

2. Iterative Clustering in the Consensus Space

We propose a new algorithm for robust multi-instance model fitting that combines the advantages of state-of-the-art algorithms and, also, follows a new formulation that does not require crisp point-to-model assignments for finding the dominant model instances.

2.1. Idea and Schematic Algorithm

The proposed method is motivated by two observations about the nature of multi-model fitting problems. First, even though all of the state-of-the-art algorithms \cite{1,3,4,24,26,34,40,63} formalize the problem as a clustering where a set of data points (cluster) represents a model instance, this assumption is incorrect in a number of real-world scenes. Moreover, one of the primary reasons of multi-model fitting algorithms often being fairly slow stems from the optimization techniques, \textit{e.g.} $\alpha$-expansion in PEARL \cite{24}, needed to solve the point-to-model assignment problem.

Our second observation is that multi-model fitting can usually be rephrased as the problem of finding multiple \textit{dominant} instances that are reasonably \textit{different}. Ideally, a \textit{dominant} instance is one that represents a real structure. Since this is not an algorithmically measurable property, we define being dominant as having a reasonably large support not shared with other dominant instances. We consider instances \textit{different} if they are “far” on the model manifold.
as proposed in Multi-X [3]. This simple formulation allows us to avoid applying complex procedures finessing to interpret point-point, model-model, and point-model interactions. Also, it further relaxes the greedy nature of the progressive model proposal strategy introduced in Prog-X [3] that enables to discover the data gradually. The pseudo-code of formalizing the multi-model problem as finding different dominant model instances is as follows:

Input: \( P \) – data points  
Output: \( I \) – model instances  
\[ I \leftarrow \emptyset \]  
\[ \text{while} \) \( \neg \)\text{Terminate}() \( \text{do} \]  
\[ I \leftarrow I \cup \text{FindDominantInstances}(P) \]  
\[ \text{while} \) \( \neg \)\text{Convergence}() \( \text{do} \]  
\[ I \leftarrow \text{SelectUniqueInstances}(I) \]  
\[ I \leftarrow \text{ImproveParameters}(I, P) \]  

2.2. Finding Dominant Model Instances

Given a set of data points \( P \) and a set of dominant model instances \( I \) proposed in earlier iterations, the objective is to find a new dominant model instance \( h \in \mathbb{R}^d \) which should be included in \( I \), where \( d \in \mathbb{R} \) is the model dimension. In the first iteration, \( I = \emptyset \).

To do so, we start similarly to RANSAC by first drawing a random sample \( S \) of data points. This is done by a state-of-the-art sampler, e.g., PROSAC [11] or P-NAPSAC [6]. Model instance \( h \) is estimated from sample \( S \). In order to decide about \( h \) being dominant or not, we define model quality function \( Q : \mathbb{R}^d \times P^* \times \mathbb{R} \rightarrow \mathbb{R} \) similarly as [4] to be calculated from the inliers of \( h \) not shared with other instances in \( I \), where \( P^* \) is the power set of \( P \). Considering the RANSAC-like inlier counting, the implied quality is

\[
Q_{\text{RSC}}(h, P, \epsilon) = \sum_{p \in P} [\phi(h, p) < \epsilon \wedge \phi(I, p) \geq \epsilon], \quad (1)
\]

where \( \epsilon \in \mathbb{R} \) is the inlier-outlier threshold and \( \phi(I, p) = \min_{h \in I} \phi(h, p) \) is the minimal point-to-model residual of point \( p \) given the kept set of dominant instances \( I \). In order to use the recent advances of RANSAC, e.g., the loss function of MAGSAC++ [6] the currently most accurate method according to a recent survey [31], \( Q_{\text{RSC}} \) is reformulated considering a continuous loss function \( f \). For practical reasons, we consider losses returning a value in-between 0 and 1. The implied quality function is

\[
Q_f(h, P, \epsilon) = |P| - \sum_{p \in P} \max (f(h, p), 1 - f(I, p)), \quad (2)
\]

where \( f(I, p) = \min_{h \in I} f(h, p) \) is the minimum loss of point \( p \) given the set of kept instances \( I \). It can be easily seen that this quality function returns high score to those instances which do not share inliers with any of the instances from \( I \). Otherwise, the quality is reduced according to the number and residuals of the inliers shared.

To determine whether instance \( h \) is dominant, we introduce parameter \( q_{\min} \), and all model instances are considered dominant where \( Q_f(h, P, \epsilon) \geq q_{\min} \). This constraint can be interpreted as a lower bound for the number of perfectly fitting data points which are not shared with any of the instances from the maintained set in \( I \).

2.3. Clustering in the Consensus Space

The next step of the algorithm, after a set \( I \) of dominant model instances have been found, is to select a subset of \( I \) consisting of instances that represent different model instances and not noisy observations of the same one. We define a model-to-model residual function \( \psi : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R} \) measuring the distance of two model instances.

Model-to-model residual. Defining a model-to-model residual function is a challenging problem. In the Multi-X algorithm [3], it was proposed to convert the model instances to point sets. The distance of two instances is the Hausdorff distance [43] of the point sets representing them. This solution is however challenging, since the conversion of geometric models to point sets in a robust manner is unclear in most of the cases. Even for homographies, there is a number of cases when this approach simply does not work, see Fig. 4 for examples. Instead, we follow the strategy proposed in the T-Linkage algorithm [32] to measure the model-to-model residual as the Tanimoto distance of the preference vectors [57] as follows:

\[
f_T(v_a, v_b) = \frac{\langle v_a, v_b \rangle}{||v_a||^2 + ||v_b||^2 - \langle v_a, v_b \rangle} . \quad (3)
\]

The preference vector of a model instance \( h \) is \( v \in [0, 1]^n \), where \( n \) is the number of input data points. Its ith coordinate is calculated as \( v_i = 1 - f(h, p_i) \), where \( f \) is the same loss function as what is used in the previous section and \( p_i \) is the ith point from \( I \). Briefly, \( v_i \) is zero if the point-to-model residual is greater than the inlier-outlier threshold. Otherwise, it is from interval \((0, 1]\). In this case, the Tanimoto distance measures the overlap of the inliers sets of two model instances where the inlier assignment is done in a smooth manner. We will call the domain of preference vectors consensus space in the rest of the paper.

Note that, while the Tanimoto distance is not a proper metric over general vector spaces, it becomes one when the preference vector is \((0, +\infty)^n\) [30]. This property holds in our case. Also note that for 0 vectors, the distance is undefined. In our case, this never happens since each model is fit to a minimal sample of \( m \) (degrees of freedom) data points which consequently have 0 residuals. Thus, at least \( m \) elements of each preference vector are 1.

\(^1\text{Such } q_{\min} \text{ parameter is often used, e.g., in Structure-from-Motion algorithms (COLMAP uses } q_{\min} = 15 \text{ [52]).}\)
finds all sought instances, the error originates from points assigned to the wrong instance. The selected scenes are the ones with the most accurate results are shown. In (a–b), the worst and best results are identical. In (c–d), the difference is negligible. The proposed method determines by assigning each point to one of the instances returned by the proposed algorithm with the minimum point-to-model residual. Black points are outliers. For each image, the highest (ME) terminated by assigning each point to one of the instances returned by the proposed algorithm with the minimum point-to-model residual.

Even though it would be straightforward to use the density modes, e.g. as in [13], it requires doing operations with the preference vectors, e.g., averaging. However, such operations are undefined in the consensus space – the average of two vectors is not necessarily the preference vector of the average instance. Thus, we replace each cluster with one of its elements that has the highest quality $Q_f$ and, thus, is the most likely to represent the sought model parameters.

In the implementation, we use the DBSCAN [15, 53] density-based clustering that runs swiftly on our problem and returns accurate solutions. DBSCAN requires two parameters, i.e., the minimum size $c_{\text{min}}$ of a cluster to be kept and a threshold $\epsilon_T$ to decide if two model instances are neighbors in the consensus space. The minimum size $c_{\text{min}} = 1$ since single-element clusters also contain dominant model instances and, thus, should be kept. The setting of threshold $\epsilon_T$ is intuitive. Setting $\epsilon_T$ to 0 means that we consider models neighbors if and only if their preference vectors are exactly the same. Parameter $\epsilon_T = 1$ means that all methods are neighbors even if they do not share inliers.

2.4. Improving Instance Parameters

In order to improve the parameters of the instances kept by the clustering algorithm, we apply an iteratively re-weighted least-squares approach starting from the initial instance parameters. We use the robust MAGSAC++ weights.

The model optimization and clustering are applied repeatedly since during the optimization step two instances might become similar and, thus, should be put in the same cluster. This iteration stops when only one-element clusters are returned by the applied clustering algorithm.

2.5. Termination Criterion

To decide when the algorithm should terminate, we use the criterion proposed in [4] that is $n_i = (|P| - |\mathcal{Z}|) \sqrt{1 - \frac{\mathcal{Z}}{\mu}} \leq m + 1$, where $\mu$ is the required confidence in the results typically set to 0.95 or 0.99; $k$ is the number of iterations; $m$ is the size of the minimal sample; $n_i$ and $|P|$ are the number of inliers and points; and $|\mathcal{Z}|$ is the cardinality of the united inlier sets of the kept model instances. This criterion is triggered if the probability of hav-
Algorithm 1 Connected Component Sampler: the next $S$.

Input: $r, r_{\min}, r_{\max}, n_{\text{steps}}$ — current, min., max. neighborhood radius and partition number;
$\mathcal{P}$ — data points; $\mathcal{A}$ — neighborhood-graph; $m$ — sample size.

Output: $S$ — sample

if $\neg\text{Initialized}(\mathcal{A})$ then $\triangleright$ Run only once
  $\mathcal{A} \leftarrow \text{BuildNeighborhood}(\mathcal{P}, r_{\max})$ $\triangleright$ Radius is $r_{\max}$
  $r \leftarrow r_{\min}$ $\triangleright$ The max. radius in $\mathcal{A}$ for the next step
  $\mathcal{C} \leftarrow \text{GetConnectedComponents}(\mathcal{A}, r)$
while $\text{Empty}(\mathcal{C}) \land r \leq r_{\max}$ do
  $r \leftarrow r + (r_{\max} - r_{\min}) / n_{\text{steps}}$
  $\mathcal{C} \leftarrow \text{GetConnectedComponents}(\mathcal{A}, r)$
  if $\neg\text{Empty}(\mathcal{C})$ then
    repeat $\triangleright$ Get the next largest dominant instance
      $S \leftarrow \text{GetLargest}(\mathcal{C}), \mathcal{C} \leftarrow \mathcal{C} \setminus \text{GetLargest}(\mathcal{C})$
    until $|S| \geq m \lor \text{Empty}(\mathcal{C})$
  if $|S| < m$ then
    $S \leftarrow \text{PROSAC}(\mathcal{P}, m)$


3. Connected Component Sampling

There have been a number of algorithms proposed, e.g. PROSAC [11], P-NAPSAC [6], to find samples that consist of data points stemming from the same model instance early. When fitting multiple instances to real-world data, it usually is a reasonable assumption that the points form spatially coherent structures [2, 6, 24, 38]. We propose a deterministic sampling that returns the connected components in a progressively densified neighborhood-graph as samples. The algorithm is shown in Alg. 1.

The user-defined parameters are the minimum ($r_{\min}$) and maximum ($r_{\max}$) neighborhood radii and the number of steps when densifying the graph ($n_{\text{steps}}$). As initialization, the method first builds neighborhood-graph $\mathcal{A}$ using the maximum radius. Then the connected components are selected from a sub-graph of $\mathcal{A}$ where all edges are ignored that are larger than the current radius $r$. This is done to avoid building $\mathcal{A}$ multiple times. The algorithm returns the largest connected component that has at least $m$ points. If there is no such component, it increases the neighborhood size by changing $r$. Note that the returned sample $S$ is not necessarily a minimal sample. If $r$ exceeds $r_{\max}$, there are no reasonable structures and, thus, it starts sampling from all data points in a global manner by the PROSAC sampler. Also note that while PROSAC is a safe-guard for cases where the data is not spatially coherent, it was never executed in experiments of Sec. 4.

4. Experimental Results

Implementation Details. The proposed method is implemented in C++ using the Eigen library and the solver implementation from the GC-RANSAC [2] repository. We combine the algorithm with a number of components from USAC [42]. The included components are the following. Sample degeneracy. The degeneracy tests of minimal samples are for rejecting clearly bad samples to avoid the sometimes expensive model estimation. For homographies, samples consisting of collinear points are rejected. Sample cheirality. The test is for rejecting samples based on the assumption that both cameras observing a 3D surface must be on its same side. For homography fitting, we check if the ordering of the four point correspondences (along their convex hulls) in both images are the same. Model degeneracy. The purpose of this test is to reject models early to avoid verifying them unnecessarily. For $F$ matrices, DEGENSAC [12] is applied to determine whether the epipolar geometry is affected by a dominant plane.

Parameters. Model-to-model threshold $\epsilon_T = 0.8$. This can roughly be interpreted as considering two model instances similar if more than 20% of their inliers are shared. The minimum quality $q_{\min} = 20$ and confidence $\mu = 0.99$. The parameters of the sampler are radii $r_{\min} = 20, r_{\max} = 200, n_{\text{steps}} = 5$. For point correspondences, the neighborhood is built on the joint 4D coordinate space. These parameters are used in all tested problems and on all datasets. Additional explanation of the hyper-parameters is in the supp. material.

4.1. Standard Benchmarks

To evaluate the proposed method on real-world problems, we use a number of publicly available datasets for homography, two-view motion, and motion fitting. The error is the misclassification error (ME), i.e., the ratio of points assigned to the wrong cluster. The proposed method
Table 1. Avg. misclassification errors (in %; 5 runs on each scene), their std. and the run-times (secs) on two-view motion and homography fitting on the AdelaideRMF dataset [66], and motion fitting on the Hopkins dataset [59]. All methods use fixed parameters. For CONSAC, we report the times of running it on GPU and CPU. The second column (|Z| needed) is “yes” for methods requiring the number of instances to fit. In the first row, the proposed method runs the P-NAPSAC [6] sampler. In the second one, the proposed CC sampler is used.

<table>
<thead>
<tr>
<th></th>
<th>Adelaide: Two-view motions</th>
<th>Adelaide: Homographies</th>
<th>Hopkins: Motions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>19 scenes</td>
<td>19 scenes</td>
<td>155 scenes</td>
</tr>
<tr>
<td>Proposed</td>
<td>no</td>
<td>5.3</td>
<td>3.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.4</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>0.11</td>
</tr>
<tr>
<td>Proposed (CC)</td>
<td>no</td>
<td>5.0</td>
<td>5.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.4</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.02</td>
<td>0.11</td>
</tr>
<tr>
<td>P-NAPSAC [6]</td>
<td>no</td>
<td>10.7</td>
<td>6.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8.7</td>
<td>5.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.52</td>
<td>1.03</td>
</tr>
<tr>
<td>Multi-X [3]</td>
<td>no</td>
<td>17.1</td>
<td>8.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12.2</td>
<td>5.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.52</td>
<td>1.03</td>
</tr>
<tr>
<td>PEARL [24]</td>
<td>no</td>
<td>29.5</td>
<td>15.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.8</td>
<td>6.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.94</td>
<td>2.61</td>
</tr>
<tr>
<td>RPA [33]</td>
<td>yes</td>
<td>17.1</td>
<td>23.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11.1</td>
<td>13.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10.24</td>
<td>622.87</td>
</tr>
<tr>
<td>RansaCov [34]</td>
<td>yes</td>
<td>55.6</td>
<td>66.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12.4</td>
<td>18.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.33</td>
<td>17.69</td>
</tr>
<tr>
<td>T-linkage [32]</td>
<td>no</td>
<td>46.7</td>
<td>54.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15.6</td>
<td>22.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.69</td>
<td>57.84</td>
</tr>
<tr>
<td>MLink [35]</td>
<td>no</td>
<td>8.6</td>
<td>5.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4.7</td>
<td>1.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.75</td>
<td>47.75</td>
</tr>
<tr>
<td>CONSAC [26]</td>
<td>no</td>
<td>–</td>
<td>5.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>–</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>–</td>
<td>8.1 / 21.0</td>
</tr>
</tbody>
</table>

We report the times of running it on GPU and CPU. The second column (|Z| needed) is “yes” for methods requiring the number of instances to fit. In the first row, the proposed method runs the P-NAPSAC [6] sampler. In the second one, the proposed CC sampler is used.

To test the proposed connected component-based sampler, we applied the proposed method with P-NAPSAC [6] and the proposed Connected Component Sampler, both of them exploiting the spatial nature of geometric data. We chose P-NAPSAC as a competitor, since it has a similar procedure, finding local structures by randomly sampling from gradually growing neighborhoods. The major difference between them is that P-NAPSAC is randomized and returns minimal samples, while the proposed Connected Component Sampler is deterministic and proposes larger-than-minimal samples as well.

Examples of multi-homography and two-view motion fitting are in Fig. 3. We chose the scenes from the AdelaideRMF dataset [66] dataset with the most ground truth models to be found. Color denotes the point-to-model assignment needed avg. std. time avg. std. time avg. std. time.

Two-view motion fitting is tested on the AdelaideRMF motion dataset consisting of 19 image pairs and correspondences manually assigned to two-view motion matrices. In this case, multiple F matrices are to be found. For the proposal step, we used the 7PT algorithm [20]. In the IRLS fitting, we applied the norm. 8PT solver [19].

The avg. errors over five runs and their std. are shown in the left block of Table 1. The proposed method leads to state-of-the-art accuracy with both tested samplers. The proposed Connected Components Sampler (CC) improves both the accuracy and processing time. The proposed method with CC is twice as accurate as the second best competitor (MLink) while being two orders-of-magnitude faster than the second fastest method (Multi-X). The proposed method runs in real-time on these scenes. On avg., out of the 45 motions in the dataset, the proposed method does not find 2 instances while returning 1 false positive.

Homography fitting is tested on the AdelaideRMF H dataset [66]. It consists of 19 image pairs with ground truth correspondences assigned manually to Hs. In these tests, we also included the errors of CONSAC [26]. Since the run-times are not reported in [26], we re-ran the algorithm both on GPU and CPU and calculated the avg. times.

We used the norm. 4PT algorithm both in the proposal and IRLS steps. The results are shown in the middle block of Table 1. The proposed method is almost twice as accurate as the second best one (CONSAC) while being significantly faster than all algorithms. It leads to the most accurate solutions while being the fastest. In this case, P-NAPSAC sampler leads to the best results. Out of the 52 Hs, the proposed method does not find 2 with 2 false positives.

Motion segmentation is tested on 155 videos of the Hopkins dataset [59]. It consists of 155 sequences divided into three categories: checkerboard, other, and traffic. The trajectories are inherently corrupted by noise, but no outliers are present. Motion segmentation in videos is the retrieval of sets of points undergoing rigid motions in a dynamic scene captured by a moving camera. It can be considered a subspace segmentation under the assumption of affine cameras. For such cameras, all feature trajectories associated with a single moving object lie in a 4D linear subspace in R2F, where F is the frame number [59].

The results are shown in the right part of Table 1. The proposed method leads to the lowest errors. It still runs in real-time. In this case, we used uniform sampling since building a neighborhood-graph (required both by the CC sampler and P-NAPSAC) on point trajectories is not trivial.
Table 2. Relative rotation $\epsilon_R$ and translation $\epsilon_p$ errors (°) on 435,190 image pairs from the 1DSfM dataset obtained by E matrix estimation; calculating E from the inliers of homographies (E from $H_8$); pose averaging on the poses decomposed from E and multiple $H_8$s; and selecting the pose with the most inliers from the decomposed ones (Pose selection) with the proposed sampler (CC).

<table>
<thead>
<tr>
<th>Method</th>
<th>$\epsilon_R$ avg.</th>
<th>$\epsilon_R$ med.</th>
<th>$\epsilon_p$ avg.</th>
<th>$\epsilon_p$ med.</th>
</tr>
</thead>
<tbody>
<tr>
<td>E matrix</td>
<td>9.51</td>
<td>3.46</td>
<td>18.15</td>
<td>9.08</td>
</tr>
<tr>
<td>E from $H_8$</td>
<td>9.56</td>
<td>3.47</td>
<td>18.21</td>
<td>9.09</td>
</tr>
<tr>
<td>Pose averaging</td>
<td>8.71</td>
<td>3.69</td>
<td>34.34</td>
<td>25.27</td>
</tr>
<tr>
<td>Pose selection</td>
<td>8.33</td>
<td>3.34</td>
<td>17.84</td>
<td>8.92</td>
</tr>
<tr>
<td>Pose selection (CC)</td>
<td><strong>8.24</strong></td>
<td><strong>3.31</strong></td>
<td><strong>17.81</strong></td>
<td><strong>8.89</strong></td>
</tr>
</tbody>
</table>

Table 3. Rotation and position errors of the global SfM implemented in [56] when initialized with a poses estimated from E matrices, and via the proposed pose selection from E and multiple $H_8$s.

<table>
<thead>
<tr>
<th>Method</th>
<th>$\epsilon_R$ avg.</th>
<th>$\epsilon_R$ med.</th>
<th>$\epsilon_p$ avg.</th>
<th>$\epsilon_p$ med.</th>
</tr>
</thead>
<tbody>
<tr>
<td>E matrix</td>
<td>11.15</td>
<td>6.58</td>
<td>10.25</td>
<td>8.93</td>
</tr>
<tr>
<td>E + mult. $H_8$s</td>
<td>7.93</td>
<td>6.21</td>
<td>10.52</td>
<td>4.60</td>
</tr>
<tr>
<td>E + mult. $H_8$s (CC)</td>
<td><strong>5.56</strong></td>
<td><strong>5.61</strong></td>
<td><strong>9.57</strong></td>
<td><strong>3.99</strong></td>
</tr>
</tbody>
</table>

4.2. Application: Relative Pose Estimation

In this section, we focus on improving relative pose estimation by exploiting multiple homographies. Pose estimation is a fundamental problem in a number of popular methods, e.g., in Structure-from-Motion algorithms. While the usual procedure to estimate a relative pose uses epipolar geometry, it is well-known that the pose can also be obtained from a homography if the cameras are calibrated. However, in most pipelines, homographies are used only if the scene is degenerate for fundamental matrix estimation, e.g., a single plane dominates the scene [12] or the camera undergoes purely rotational motion. In this section, we aim to propose a way of exploiting multiple homographies to improve the relative pose accuracy. See Fig. 5 for an example.

We downloaded the 1DSfM dataset [64] and applied COLMAP [52] to obtain a reconstruction that can be used as ground truth. Note that the 1DSfM dataset provides a ground truth, however, it was created by the Bundler algorithm [54] that is more than 10 years old. We use the following approach in order to find potentially matching image pairs. First, we extract GeM [41] descriptors with ResNet-50 [21] CNN, pre-trained on GLD-v1 dataset [39]. Then we calculate the inner-product similarity between the descriptors, resulting in an $n \times n$ similarity matrix. In the experiments, we use only the image pairs with similarity higher than 0.4 [5]. Finally, we estimated multiple homographies for all considered image pairs, 434,587 in total.

We tested the following approaches to recover the relative pose from multiple $H_8$s:

2. Decomposing all found homographies and, also, the essential matrix relative poses and running pose averaging by [10, 65].
3. Decomposing each homography [36] and, also, the essential matrix to pose and selecting the one which has the most inliers determined by thresholding the re-projection error. The translation is then re-estimated by solving equation $p^B_3(t) R p_t = 0$ with known rotation $R$, where $[t]_B$ is the cross-product matrix of translation $t$ and $[t]_B R$ is the essential matrix. Details are in the supplementary material.

The results are reported in Table 2. Results when using the proposed connected component sampler (CC) are also shown. To measure the error in the rotation, we calculate the angular difference between the ground truth $R$ and estimated $R$ ones as $\epsilon_R = \cos^{-1}(\text{tr}(R R^T) - 1)/2$. Since the translation is up to scale, the error is the angular difference $\epsilon_t$ of the ground truth and estimated translations. The avg. rotation and translation errors are improved by, respectively, 1.27 and 0.34 degrees compared to E estimation. CC sampler leads to the best results. Since it is extremely fast, the computational overhead is merely a few ms.

We applied the global SfM implemented in the Theia library [56] initialized with the poses estimated in the proposed way and, also, with the poses estimated using only essential matrices. The accuracy of the reconstruction is reported in Table 3. We report the average rotation (avg. $\epsilon_R$, in degrees) and position errors (avg. $\epsilon_p$, in meters) and, also, the median errors averaged over the scenes. The proposed algorithm with the CC sampler significantly reduces both the rotation and position errors of the reconstruction.

4.3. Application: Fast-moving Object Detection

In this section, we estimate the trajectories of objects that are significantly blurred by their motion. As defined in [47], an image $I$ of such blurred object is formed as a composition of the blurred object appearance and the background

$$I = H \ast F + (1 - H \ast M) B,$$  

where the sharp object appearance $F$ with mask $M$ encodes the object, blur kernel $H$ encodes the trajectory, and $B$ rep-
resolves the background. Input image $I$ and background $B$ are assumed to be known. The unknowns in (4) are estimated either by alternating energy minimization with additional priors [27–29,45,46,48] or more recently by learning from synthetic data [49,50] and neural rendering [51].

The formation model in (4) encodes the trajectory by the blur kernel. However, there are no guarantees that the blur kernel corresponds to a physically plausible trajectory, which is assumed to be piece-wise linear due to bounces. Blur kernels also contain other responses due to other moving objects in the scene. In the extreme case, if two fast-moving objects intersect or fly close to each other, the blur kernel will contain multiple responses corresponding to each motion. In practice, the estimated blur kernels are noisy, with many outliers, and contain artifacts due to shadows, low contrast, and discretization. Motion blur priors [61] have been proposed to reduce these issues, but extracting the final continuous trajectory is still a challenging multi-instance model fitting task (see Fig. 6 for examples).

Recent methods [28,48] address this task by employing Sequential RANSAC [25,60] on the thresholded blur kernels. We extract blur kernels using the TbD method [28] from all sequences in the TbD [28] and TbD-3D [48] datasets. The TbD dataset is simpler since it contains mostly uniformly colored objects moving in the plane parallel to the camera plane. The TbD-3D dataset is more challenging with highly textured objects that are rotating and moving in 3D. The ground truth sub-frame object location is given from a high-speed camera. We estimate multiple line segments in each blur kernel and measure the average $L_2$ distance of each ground truth location to the closest fitted line segment. Table 5 shows the average error, its standard deviation, and average run-time for a wide range of state-of-the-art methods. We used the implementations provided by the authors. The proposed method outperforms all compared algorithms both in terms of accuracy and processing time, running in real-time. Additional results, e.g. demonstrating the effect of the proposed soft assignment, are in the supplementary material. Without considering soft assignment, continuous chains can not be found. This leads to losing short segments and affects the accuracy notably.

### 4.4. Pose from Generalized Camera

To further test the pose selection technique from multiple homographies and an essential matrix, as proposed in Section 4.2, we downloaded the KITTI odometry dataset [17], where each frame consists of the images of two cameras. We considered the two cameras as a generalized one and estimated the pose between this camera and the left image of the next frame. We used the generalized essential matrix [70] (E4+2) and homography [7] (H3+2) solvers. For finding a single E4+2 or H3+2, we used GC-RANSAC [2]. The methods were tested on a total of 23 190 frame pairs.

The results are in Table 4. The proposed technique (E + mult. Hs), selecting the best pose from the set decomposed from an essential matrix and multiple homographies, leads to the most accurate results in terms of average rotation and position errors. Its median rotation error is similar to H3+2. Its median position error is the lowest.

### 5. Conclusion

We propose a new multi-instance model fitting algorithm that is a simple iteration of instance proposal, clustering in the consensus space, and parameter re-estimation. Due to not forming crisp point-to-model assignments, the method runs in real-time on a number of vision problems. On two-view motion estimation, it is at least two orders-of-magnitude faster than the competitors. It leads to results superior to the state-of-the-art both in terms of accuracy and run-time on the standard benchmark datasets. Moreover, the proposed Connected Component sampler outperforms the recent P-NAPSAC on a number of real-world problems. In addition, we demonstrated on a total of 458 569 images or image pairs that using multiple model instances, e.g. homographies or line segments, is beneficial for various popular vision applications, e.g., Structure-from-Motion.

Acknowledgements. This project was supported by the EIT Postdoc fellowship, by the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" (RCI), and by the Robert Bosch GmbH. The conference presentation was supported by Czech Technical University in Prague (RCI project).

---

**Table 5.** The avg. and std. accuracy (px) and run-time (secs) of multiple line segment detection for finding the trajectories of fast-moving objects. The number of images are in brackets.


[69] Qing Zhao, Yun Zhang, Qianqing Qin, and Bin Luo. Quantized residual preference based linkage clustering for model selection and inlier segmentation in geometric multi-model fitting. Sensors, 20(13):3806, 2020. 2
