Multibody motion segmentation using the geometry of 6 points in 2D images

Klas Nordberg
Computer Vision Laboratory
Department of Electrical Engineering
Linköping University, Sweden
Email: klas@isy.liu.se

Vasileios Zografos
Computer Vision Laboratory
Department of Electrical Engineering
Linköping University, Sweden
Email: zografos@isy.liu.se

***Abstract*—We propose a method for segmenting an arbitrary number of moving objects using the geometry of 6 points in 2D images to infer motion consistency. This geometry allows us to determine whether or not observations of 6 points over several frames are consistent with a rigid 3D motion. The matching between observations of the 6 points and an estimated model of their configuration in 3D space, is quantified in terms of a geometric error derived from distances between the points and 6 corresponding lines in the image. This leads to a simple motion inconsistency score, based on the geometric errors of 6 points that in the ideal case should be zero when the motion of the points can be explained by a rigid 3D motion. Initial point clusters are determined in the spatial domain and merged in motion trajectory domain based on this score. Each point is then assigned to the cluster, which gives the lowest score. Our algorithm has been tested with real image sequences from the Hopkins155 database with very good results, competing with the state of the art methods, particularly for degenerate motion sequences. In contrast to the motion segmentation methods based on multi-body factorization, that assume an affine camera model, the proposed method allows the mapping from 3D space to the 2D image to be fully projective.**

***Index Terms*—motion segmentation; 6-point geometry; fully projective camera model; Hopkins 155**

I. Introduction

Motion segmentation is the problem of segmenting the image, as part of an image sequence, into objects where each object is moving with a distinct 3D motion pattern. Given a set of interest points on the imaged objects, each interest point generates a trajectory over the image sequence, and the segmentation is performed by analyzing these trajectories. There exist a plethora of methods for solving the segmentation problem, and as a result, several ways of grouping these methods based on common principles of operation and underlying assumptions.

The two current state-of-the-art methods, relative to standard datasets such as Hopkins155 [1], are Sparse Subspace Clustering (SSC) [2] and Spectral Curvature Clustering (SCC) [3]. Both methods take as input a set of points sampled from a mixture of affine or linear subspaces, one for each object to be determined. The subspaces are determined by clustering the samples based on sparse representations [2] or by iteratively clustering the samples based on spectral curvature [3]. Other common methods in the literature are based on Multi-Stage unsupervised Learning (MSL) [4], Local Subspace Affinity (LSA) [5], Agglomerative Lossy Compression (ALC) [6], General Principal Component Analysis (GPCA) [7], or on RANdom SAmple Consensus (RANSAC) [8].

II. Mathematical background

We consider a set of six 3D points, with homogeneous coordinates $\mathbf{x}_k$, projected onto an image according to the pinhole camera model: $\mathbf{y}_k \sim \mathbf{C}\mathbf{x}_k, k = 1, \ldots, 6$, where $\mathbf{y}_k$ are the corresponding homogeneous image coordinates, $\mathbf{C}$ is the 3×4 camera matrix, and $\sim$ denotes equality up to a scalar multiplication. $\mathbf{x}_k$ vary over time, and we assume that this variation can be modeled as a 3D homography transformation[1], that is

$$\mathbf{y}_k \sim \mathbf{C}\,\mathbf{H}\,\mathbf{x}_k, \quad k = 1, \ldots, 6, \tag{1}$$

where $\mathbf{H}$ is a time dependent 3D homography that transforms the set of 3D points from some reference configuration to the specific observation that produces the image coordinates $\mathbf{y}_k$ that now implicitly depend on $\mathbf{H}$. The problem addressed here is how we can determine if an observed set of image points $\mathbf{y}_k$ really is given by (1) for a particular set of 3D points $\mathbf{x}_k$, the *reference 3D points*, but with $\mathbf{C}$ and $\mathbf{H}$ unknown.

To a large extent this problem has been solved by Quan [10], later extended in [11], [12], [13], and we summarize here the main results that will be used as a basis for the remaining parts of this paper. In general, the homogeneous coordinates of the 3D points can be transformed by a suitable 3D homography $\mathbf{H}_x$ to *canonical* homogeneous 3D coordinates $\mathbf{x}' = \mathbf{H}_x\,\mathbf{x}$, where:

$$(\mathbf{x}'_1\ \mathbf{x}'_2\ \mathbf{x}'_3\ \mathbf{x}'_4\ \mathbf{x}'_5\ \mathbf{x}'_6) \sim\sim \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & X \\ 0 & 1 & 0 & 0 & 1 & Y \\ 0 & 0 & 1 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 & 1 & T \end{pmatrix}. \tag{2}$$

Here $\sim\sim$ denotes equality up to an individual scalar multiplication on each column. Similarly, for the particular observation

[1] 3D homography transformations, or projective transformations, are general non-singular linear transformations on the homogeneous 3D coordinates, including rigid transformations of the 3D space [9].

of the corresponding image points we can transform them to canonical homogeneous 2D coordinates $\mathbf{y}'_k = \mathbf{H}_y\,\mathbf{y}_k$, where

$$(\mathbf{y}'_1\ \mathbf{y}'_2\ \mathbf{y}'_3\ \mathbf{y}'_4\ \mathbf{y}'_5\ \mathbf{y}'_6) \sim\sim \begin{pmatrix} 1 & 0 & 0 & 1 & u_5 & u_6 \\ 0 & 1 & 0 & 1 & v_5 & v_6 \\ 0 & 0 & 1 & 1 & w_5 & w_6 \end{pmatrix}. \quad (3)$$

$\mathbf{H}_x$ and $\mathbf{H}_y$ depend on $\mathbf{x}_1, \ldots, \mathbf{x}_5$ and $\mathbf{y}_1, \ldots, \mathbf{y}_4$, respectively, and $\mathbf{y}'_k \sim \mathbf{H}_y\,\mathbf{C}\,\mathbf{H}\,\mathbf{H}_x^{-1}\,\mathbf{x}'_k$.

The main result in [10] is that from these transformed coordinates we can compute a set of five *relative invariants* of the image points, denoted i_k, and of the 3D points, denoted $\tilde{I}_k$, according to:

$$\mathbf{z} = \begin{pmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \\ i_5 \end{pmatrix} = \begin{pmatrix} w_6(u_5 - v_5) \\ v_6(w_5 - u_5) \\ u_5(v_6 - w_6) \\ u_6(v_5 - w_5) \\ v_5(w_6 - u_6) \end{pmatrix} \quad \mathbf{s} = \begin{pmatrix} \tilde{I}_1 \\ \tilde{I}_2 \\ \tilde{I}_3 \\ \tilde{I}_4 \\ \tilde{I}_5 \end{pmatrix} = \begin{pmatrix} XY - ZT \\ XZ - ZT \\ XT - ZT \\ YZ - ZT \\ YT - ZT \end{pmatrix} \quad (4)$$

such that they satisfy the constraint

$$\mathbf{z} \cdot \mathbf{s} = i_1\,\tilde{I}_1 + i_2\,\tilde{I}_2 + i_3\,\tilde{I}_3 + i_4\,\tilde{I}_4 + i_5\,\tilde{I}_5 = 0. \quad (5)$$

To realize what this means, we notice that this constraint includes scalars derived from the reference 3D coordinates $\mathbf{x}_k$ (before they are transformed) and observed image points $\mathbf{y}_k$ (after the transformation $\mathbf{H}$ is made), but neither $\mathbf{C}$ nor $\mathbf{H}$ are explicitly included. Therefore, the constraint is satisfied regardless of how we transform the 3D points, as long as they are all transformed by the same $\mathbf{H}$. As long as the observed image coordinates are consistent with (1), the corresponding relative image invariants $\mathbf{z}$ must satisfy (5) for a fixed $\mathbf{s}$ computed from the 3D reference points. The canonical transformations $\mathbf{H}_x$ and $\mathbf{H}_y$ can be conveniently included into the unknowns $\mathbf{C}$ and $\mathbf{H}$. In short, (5) provides a necessary, but not sufficient, constraint for the matching between the observed image points and the 3D reference points.

A. Novelties.

In [10] $\mathbf{s}$ and its relation to $\mathbf{z}$ were derived implicitly by observing that there exist $\mathbf{H}_x$ and $\mathbf{H}_y$ such that (2) and (3) are satisfied and then solving for the unknown camera matrix. $\mathbf{H}_x$ and $\mathbf{H}_y$ are unique and if we do the math explicitly, e.g., using symbolic tools for mathematics, we get:

$$\mathbf{z} = \alpha \begin{pmatrix} D_{126}D_{354} \\ D_{136}D_{245} \\ D_{146}D_{253} \\ D_{145}D_{263} \\ D_{135}D_{246} \end{pmatrix}, \quad \alpha = \frac{D_{123}}{D_{124}D_{234}D_{314}}, \quad (6)$$

$$D_{ijk} = (\mathbf{y}_i \times \mathbf{y}_j) \cdot \mathbf{y}_k = \det\begin{pmatrix} \mathbf{y}_i & \mathbf{y}_j & \mathbf{y}_k \end{pmatrix}. \quad (7)$$

Since $\mathbf{z}$ is linearly combined with $\mathbf{s}$ in (5), the scalar α can be omitted in the computation of $\mathbf{z}$ since we are only interested in whether or not $\mathbf{z} \cdot \mathbf{s} = 0$. This is further motivated by the fact that we compute $\mathbf{s}$ and $\mathbf{z}$ from homogeneous 3D and 2D coordinates, which are elements of the projective spaces P^3 and P^2 and therefore of undetermined scaling. This means that also $\mathbf{s}$ and $\mathbf{z}$ are projective elements, in this case of P^4.

In [11], [14] the matching constraint $\mathbf{z} \cdot \mathbf{s} = 0$ takes the form of an incidence relation between the six points and corresponding six lines. From the above expression of the elements in $\mathbf{z}$ as functions of the six points, the point-line incident relations can be derived in a straight-forward way. For example, we can rewrite (5) as $\mathbf{z} \cdot \mathbf{s} = \mathbf{l}_1 \cdot \mathbf{y}_1 = 0$ with

$$\mathbf{l}_1 = \mathbf{l}_{26}D_{354}\tilde{I}_1 + \mathbf{l}_{36}D_{245}\tilde{I}_2 + \mathbf{l}_{46}D_{253}\tilde{I}_3 + \\ + \mathbf{l}_{45}D_{263}\tilde{I}_4 + \mathbf{l}_{35}D_{246}\tilde{I}_5 \quad (8)$$

where $\mathbf{l}_{ij} = \mathbf{y}_i \times \mathbf{y}_j$. $\mathbf{l}_1$ depends on the five image points $\mathbf{y}_2, \ldots, \mathbf{y}_6$ and on $\mathbf{s}$. A similar exercise can be made for the other five image points and in general we can write the matching constraint as $\mathbf{z} \cdot \mathbf{s} = \mathbf{l}_k \cdot \mathbf{y}_k = 0$ where $\mathbf{l}_k$ depends on $\mathbf{s}$ and five image points: $\{\mathbf{y}_i, i \neq k\}$. With this description of the matching constraint it makes sense to interpret $\mathbf{l}_k$ as the dual homogeneous coordinates of a line in the image plane. To each of the 6 image points, there is a corresponding line and the constraint is satisfied iff all 6 lines intersect their corresponding image points.

The main reason for deriving the lines is that they allow us to quantify Quan's matching constraint constraint in terms of a *geometric error*, derived from the distances between the points and their corresponding lines, rather than an algebraic error. This approach is common practice in epipolar and multi-view geometry, motivated by more robust estimation of the given geometry [9] and independence of scaling factors of projective elements, e.g., α in Equation (6).

B. Estimation of $\mathbf{s}$.

By replacing each $\tilde{I}_k$ with its corresponding second order expression in X, Y, Z, T, (4), and then expanding the terms, we get an *internal constraint* on $\mathbf{s}$ in terms of a homogeneous third order polynomial in its elements that must vanish (see Appendix A in [13]):

$$\tilde{I}_1\tilde{I}_2\tilde{I}_5 - \tilde{I}_1\tilde{I}_3\tilde{I}_4 + \tilde{I}_2\tilde{I}_3\tilde{I}_4 - \tilde{I}_2\tilde{I}_3\tilde{I}_5 - \tilde{I}_2\tilde{I}_4\tilde{I}_5 + \tilde{I}_3\tilde{I}_4\tilde{I}_5 = 0. \quad (9)$$

From three observations of the 6 points we get three linear homogeneous equations $\mathbf{z}_k \cdot \mathbf{s}, k = 1, 2, 3$, in the five unknown elements of $\mathbf{s}$. This means that we can determine a 2-dimensional basis $\bar{\mathbf{s}}_1, \bar{\mathbf{s}}_2$ of the space of a possible $\mathbf{s}$ that matches the three observations, i.e., $\mathbf{s} = \gamma\,\bar{\mathbf{s}}_1 + (1 - \gamma)\,\bar{\mathbf{s}}_2$. Inserting this into Equation (9), gives a third order polynomial in γ. This leads to a method for estimating $\mathbf{s}$ from observed data in terms of six point correspondences over three images. Since γ is obtained by solving a third order polynomial, this method produces 3 different solutions $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$.

The above discussion on how to determine $\mathbf{s}$ refers to the case when the six 3D points are in general positions. A particular case of non-generality occurs when the points are in a plane. In this case, and in general, $\mathbf{z}$ is non-zero but since 3D homographies now correspond to 2D homographies in the image, and $\mathbf{z}$ by definition is invariant to such transformations,

$\mathbf{z}$ is invariant to any transformation $\mathbf{H}_x$. Consequently, the solution space for $\mathbf{s}$ described above increases to four dimensions which, however, still means that the above estimation method will provide correct solutions, only now they are not unique.

C. Matching score.

For motion analysis, these results can be used to verify the motion consistency given a set of 6 image points with known trajectories. From 3 observations of the 6 points, the method described above can be used to give $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$. If the 6 point are moving consistently, i.e., if they belong to the same object, the matching between one $\mathbf{s}_k$ and the 6 points should be small (ideally zero) over the trajectory time span.

Based on this idea, we compute a *motion inconsistency score* for set of 6 points as follows. The first three time frames of their trajectories are used for estimation of $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$. These are then matched against the trajectories by taking observation at time t to compute the six lines as described above and measure the distance from each line $\mathbf{l}_k$ to its corresponding point $\mathbf{y}_k$, and then take the maximum distance over the six lines. This is denoted $d_k(t)$, where k=1, 2, 3 are the different maximum distances given by three different solutions $\mathbf{s}_k$. Finally, the score is given by

$$S = \max_t(\min_k [d_k(t)]) \tag{10}$$

In the case that the 6 points are on a single object, one of the three $\mathbf{s}_k$ should give a good match, i.e., a small $d_k(t)$ for all t, which is the motivation for the above formulation of the score S. If the 6 points are on two or more objects however, S should instead be large, an assumption that is true depending on how different the motion patterns of the objects are.

III. Algorithm

We propose a simple yet effective algorithm that can be used for the segmentation of multiple moving objects in a scene. It involves selecting a number of 6-point clusters that will form the initial *seeds* from which the segmentation will evolve. The remaining points will be assigned to these clusters based on a simple "motion consistency", or rather "motion inconsistency", criterion.

We start with a basic clustering algorithm (in this case K-means) in the image domain to divide the points from a single frame into spatially distinct regions. These may not correspond to motion distinct regions, but this is acceptable since we will merge or discard any redundant clusters later on. At this stage we only need to obtain the approximate object centres in the image. We then select 6-points at a certain distance τ_D from each K-means centre and form our initial seed clusters. Following this, we perform a merging step in motion trajectory space so as to combine clusters that describe the same motion, and eliminate cluster redundancies. Once we have obtained the motion consistent 6-point clusters, we can evaluate the remaining points and assign them to the appropriate cluster based on the score in (10). As such, for each remaining point P_i yet to be classified, and each 6-point cluster C_j=$\{P_1^j,...,P_6^j\}$ we compute (10) from the points $\{P_i, P_2^j,...,P_6^j\}$. The rationale behind this is that if the point P_i has the same (or very similar) motion trajectory in t frames, as does the cluster C_j, then by adding P_i to C_j in this way the score should remain low. Conversely, if their respective motions are different, then their combination will lead to an inconsistency with high score. We are using a "winner takes all" approach with point P_i assigned to the cluster with the lowest score and for this reason there is no threshold associated with the actual classification stage.

A. Cluster merging and rejection.

The merging step takes us from spatial segmentation to consistent motion clusters. In order to merge two 6-point clusters C_k and C_l with $k \neq l$, we generate a small number ($\approx$ 20-30) of intermediate combination clusters C_r that contain permutations of 6 randomly chosen points from both C_k and C_l. We then calculate the scores for these intermediate clusters C_r and take their median, resulting in a merging score S_m. This is to avoid excessively high or low values influencing the merging results and producing inconsistencies. If S_m is below some threshold τ_m then the two clusters are merged. This threshold may be arbitrary selected or since the function that maps the number of merged clusters N to τ_m is monotonically decreasing, we can equally select a value for the clusters and derive τ_m. This is similar to the other competing methods in the literature that make use of explicit knowledge about the number of moving objects in the scene.

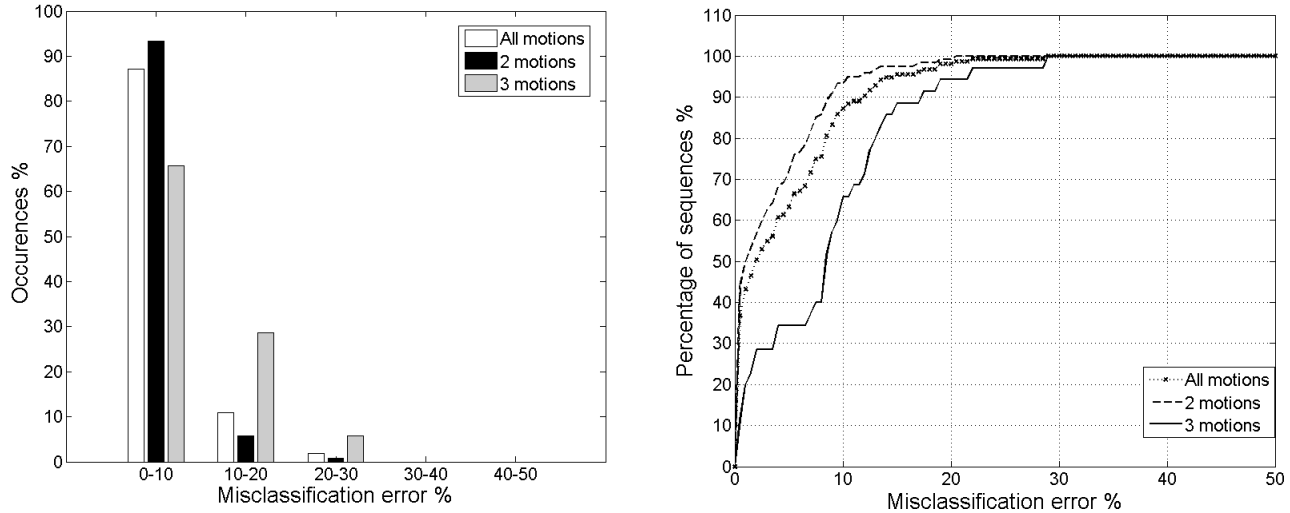
It is quite possible for a seed cluster to contain points from two or more differently moving objects. In this case however, such a cluster will describe a unique motion that no (or due to noise, very few) points will be assigned to. Such clusters can be easily detected during classification since they will grow very little in size. They will be removed and their points returned to the heap for re-classification. The algorithm is presented in pseudocode in Algorithm 1.

```
Create spatial clusters using K-means
Merge spatial clusters into C_j
foreach point P_i do
    foreach cluster C_j do
        Select 6 points {P_i, P_2^j, ..., P_6^j}.
        Calculate score S_{i,j} from (10).
    end
    Assign P_i to cluster with min(S_{i,j}).
    Reject inconsistent clusters.
end
```

Algorithm 1: Motion clustering pseudocode.

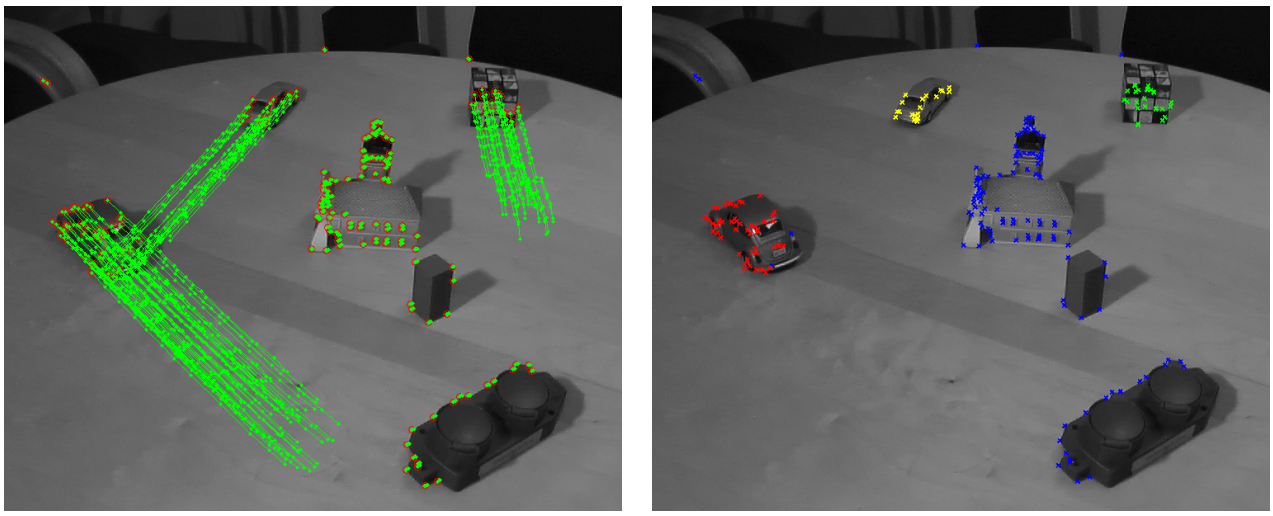
IV. Experimental results

We have carried out a number of experiments on real image sequences from the publicly available Hopkins155 database [1]. It includes different motion sequences of 2 and 3 objects, of various degrees of classification difficulty (perspective, degenerate or articulated motions) and is corrupted by tracking noise. For our experiments, we set the merging



(a) (b)

Fig. 1. Hopkins155 database results.



(a) 4 different motions (b) 0.4% resulting segmentation error

Fig. 2. 4 object motion example. Best viewed in colour.

threshold τ_m to produce 2 or 3 seed clusters, depending on each sequence in the database. Furthermore, we adjusted the minimum distance parameter τ_D to fine-tune the classification accuracy. This tuning is not essential but it can bring some small improvements on the classification results. τ_D depends on the points and K-means centres density and has to be determined by trial and error for each dataset.

Our results (six point consistency, SPC) for 2 and 3 motions are shown in Figure 1 and Tables I and II. Note also that we compare against the published results of the state-of-the-art methods. If we look at Table I, we can see that our method gives good results on average for most sequences, while being particularly effective for the degenerate traffic sequences (2nd after SSC). Again, for the articulated sequences we rank 2nd and 3rd overall for all 2 motion sequences. By comparison, our performance is close to MSL for the checkerboard data. In Table II, we see a somewhat similar behaviour with very good results on the traffic sequences and results closer to MSL for the other sets. Considering however the simplicity and speed of our method compared to the MSL and other approaches, these results are very encouraging. Finally, if we look at Figure 1 and comparing with the results published in [15], we see that our method performs at the same levels with most of the state-of-the-art, with a performance over 90% occurrences at 0-10% error rates for 2 motions and close to 65% occurrences for 3 motions. Overall, for all motions, the occurrences peak around 87% for the same error rates. However, we outperform all other methods, including SSC and SCC in the higher error regions since we have no error greater than 30%. As a result the distributions in Figure 1(a) have much heavier tails than any method in [15]. This is also mirrored in Figure 1(b) where we reach 100% on the y-axis faster than the other methods. Of

	GPCA	LSA	RANSAC	MSL	ALC	SSC	SCC	SPC
Checkerboard: 78 sequences								
Mean:	6.09	2.57	6.52	4.46	1.55	1.12	1.77	4.49
Median	1.03	0.27	1.75	0.00	0.29	0.00	0.00	3.69
Traffic: 31 sequences								
Mean:	1.41	5.43	2.55	2.23	1.59	0.02	0.63	0.22
Median	0.00	1.48	0.21	0.00	1.17	0.00	0.14	0.00
Articulated: 11 sequences								
Mean:	2.88	4.10	7.25	7.23	10.70	0.62	4.02	2.18
Median	0	1.22	2.64	0.00	0.95	0.00	2.13	0.00
All: 120 sequences								
Mean:	4.59	3.45	5.56	4.14	2.40	0.82	1.68	3.18
Median	0.38	0.59	1.18	0.00	0.43	0.00	0.07	1.08

TABLE I
THE 2 MOTION ERROR % RESULTS.

	GPCA	LSA	RANSAC	MSL	ALC	SSC	SCC	SPC
Checkerboard: 26 sequences								
Mean:	31.95	5.80	25.78	10.38	5.20	2.97	6.23	10.71
Median	32.93	1.77	26.01	4.61	0.67	0.27	1.70	9.61
Traffic: 7 sequences								
Mean:	19.83	25.07	12.83	1.80	7.75	0.58	1.11	0.73
Median	19.55	23.79	11.45	0.00	0.49	0.00	1.40	0.73
Articulated: 2 sequences								
Mean:	16.85	7.25	21.38	2.71	21.08	1.42	5.41	6.91
Median	28.66	7.25	21.38	2.71	21.08	0.00	5.41	6.91
All: 35 sequences								
Mean:	28.66	9.73	22.94	8.23	6.69	2.45	5.16	8.49
Median	28.26	2.33	22.03	1.76	0.67	0.20	1.58	8.36

TABLE II
THE 3 MOTION ERROR % RESULTS.

course we are not limited to 3 motions but can deal with an arbitrary number of moving objects as can be seen in Figure 2 (a) and (b).

Our approach is very efficient with an average running time of 0.01 sec for calculating the score in (10) for each point, using Matlab on a 2.4 GHz CPU. In addition, the algorithm is of linear complexity on the number of points$\times$clusters, a step that can very easily be separated and parallelised for faster computation.

V. CONCLUSION

We have presented a novel approach for the segmentation of a number of moving objects, using the geometry of 6 points in 2D images to infer motion consistency. The proposed method is based on Quan's [10] geometry of six points in a 2D image that allows us to determine whether or not observations of 6 points over several frames are consistent with a rigid 3D motion of the corresponding 3D points. A main feature of the proposed method is that the motion consistency is determined based on a *geometric error*, given by distances between the 6 points and corresponding 6 lines in the image. This leads to an *motion inconsistency score* that quantifies, in geometric terms, how much the motion of 6 points deviates from the case of rigid 3D motion. Another feature of this approach is that it allows a *fully projective camera model*, instead of being restricted it to the affine camera model, stipulated by the factorization-based motion segmentation methods described in the literature.

Our algorithm begins by selecting 6-point object clusters in the image domain and proceeds by merging them in the motion trajectory domain. This is done using the motion inconsistency score, where initial clusters that have sufficiently similar motions are merged. The final segmentation is carried out by assigning each remaining object point to the cluster that gives the lowest score.

We have demonstrated the efficacy of our solution on the Hopkins 155 motion segmentation dataset, with good results (especially for degenerate motions) that are comparable with many state of the art methods. In particular, we perform the best for error rates above 30% for all sequences in the database. Furthermore, out method is not restricted to 3 different motions in the scene, but rather scales quite well as the number of individual motions is increased.

ACKNOWLEDGMENTS

The research leading to these results has been funded from the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement no 215078, DIPLECS. We would like to thank Stefan Carlsson for pointing out the connection between Quan's work and an initial paper on 6 point motion consistency based on the geometric error [14].

REFERENCES

[1] P. Tron and R. Vidal, "A Benchmark for the Comparison of 3-D Motion Segmentation Algorithms," in *CVPR*, 2007.
[2] E. Elhamifar and R. Vidal, "Sparse Subspace Clustering," in *CVPR*, 2009.
[3] G. Chen and G. Lerman, "Motion Segmentation by SCC on the Hopkins 155 Database," in *ICCV*, 2009.
[4] Y. Sugaya and K. Kanatani, "Geometric Structure of Degeneracy for Multi-body Motion Segentation," in *SMVC*, 2004.
[5] J. Yan and M. Pollefeys, "A General Framework for Motion Segmentation: Independent, Articulated, Rigid, Non-rigid, Degenerate and Non-degenerate," in *ECCV*, 2006.
[6] S. R. Rao, R. Tron, E. Vidal, and Y. Ma, "Motion Segmentation via Robust Subspace Separation in the Presence of Outlying, Incomplete, or Corrupted Trajectories," in *CVPR*, 2008.
[7] R. Vidal, R. Tron, and R. Hartley, "Multiframe Motion Segmentation with Missing Data Using PowerFactorization and GPCA," *IJCV*, vol. 79, no. 1, pp. 85–105, 2008.
[8] P. H. S. Torr, "Geometric motion segmentation and model selection," *Phil. Trans. R. Society Lond. A*, vol. 356, pp. 1231–1340, 1998.
[9] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.
[10] L. Quan, "Invariants of Six Points and Projective Reconstruction From Three Uncalibrated Images," *PAMI*, vol. 17, no. 1, pp. 34–46, 1996.
[11] S. Carlsson, "Duality of Reconstruction and Positioning from Projective Views," in *Workshop on Representations of Visual Scenes*, 1995.
[12] D. Weinshall, M. Werman, and A. Shashua, "Duality of multi-point and multi-frame Geometry: Fundamental Shape Matrices and Tensors," in *ECCV*, 1996.
[13] P. H. S. Torr and A. Zisserman, "Robust parameterization and computation of the trifocal tensor," *Image and Vision Computing*, vol. 15, pp. 591–605, 1997.
[14] K. Nordberg, "Single-view matching constraints," in *International Symposium on Visal Computing*, 2007.
[15] "http://www.vision.jhu.edu/data/hopkins155/."