2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Common Field-of-View of Cameras in Robotic Swarms

Chen Zhu[1], Christoph Bamann[2], Patrick Henkel[3] and Christoph Günther[4]

Abstract— **Cooperative swarms of robots with cameras can provide stereo and multi-view vision. They are robust against failures and introduce diversity in the case of poor conditions. Cameras are used both for data acquisition and navigation. Additional sensors also play a role in the SLAM tasks (Simultaneously Localization and Mapping). The independent mobility of the robots/cameras leads to a situation in which the common field-of-view (FOV) of cameras changes continuously. The present paper addresses the task of acquiring and tracking the cameras FOVs. State-of-the-art FOV characterization techniques count feature points or do image segmentation. These methods are often not accurate or complex. We propose an adaptive common FOV detection method based on fuzzy plane clustering. The performance of the method is shown to be invariant under baseline scaling. An autonomous grouping algorithm is further proposed with respect to both distance of robots and overlapping FOV of cameras.**

I. INTRODUCTION

Autonomous robotic exploration missions usually rely on several sensors such as IMUs, laser scanners and cameras to navigate the robots. Camera subsystems provide a high level of details about the environment. They thus play a significant role in GPS denied environments or extra-terrestrial planets. VSLAM (Visual Simultaneously Localization and Mapping) techniques have shown promising performance for navigation and mapping without knowledge of both the environment and the robot's position. VSLAM uses images from stereo or monocular cameras to determine the coordinates of feature points as well as the location and attitude of the camera [1][2].

In order to increase the system robustness against hazards in the mission, e.g. getting stuck in a sandy area, and to improve the efficiency of exploration, the use of robotic swarms has been proposed [3][4]. The swarms are composed of autonomous units, such as rovers or quadrocopters, for example. The use of swarm, which move and adjust their attitude independently raises issues in the fusion of the data, however. The overlapping regions of distinct cameras vary over time. The common field-of-view (FOV) must be characterized, so that common features from different robots can be matched efficiently as a function of time. This is significant in both localization and mapping. The grouping of robots is motivated by improving the observability and efficiency. The common FOV must be known so that cameras with significant overlap of their FOV can be grouped. Snavely's state-of-the-art FOV detection approach [5] is based on counting matching feature points. Bruckner et al. [6] improved the approach by releasing the constraints on the correctness of the feature matching so that it is more robust to outliers and matching errors. Zou and Tan utilized such approaches in robotic swarm exploration, and proposed a state-of-the-art method to group, split, and merge robots according to the number of common features [7]. The result outperforms the intuitive grouping strategy according to robot distances whenever cameras are involved. The spatial distribution of the features are usually inhomogeneous and sparse. Their counting is not much related to the size of the FOV. Additionally, the baseline used for the triangulation of features is not accurately known. Thus, the FOV characterization should not be sensitive to errors in the baseline. Defining such a characterization is the main aim of the present paper.

In Section II, we develop a reliable characterization of the FOV of two cameras. It characterizes the regions seen by several cameras. The method is based on the adaptive fuzzy clustering of planes. We show that the resulting algorithm is invariant under changes in the baseline length. The FOV detection scheme is the basis for an autonomous grouping algorithm developed in Section III. It utilizes spectral clustering technique, in which the associated metric includes the common area. In Section IV, we provide simulation results for both the FOV detection algorithm and the grouping strategy. Both algorithms outperform state-of-the-art approaches.

II. ADAPTIVE COMMON FIELD-OF-VIEW DETECTION

The automatic detection of overlapping regions in images of a scene taken by several cameras is difficult, even if the positions and orientations of the cameras are known. The overlapping region is both a function of the angle and distance of the object plane, as illustrated in Fig. 1.

In many environments, the assumption that most feature points are distributed on nearly planar surfaces or facets is valid. In some environments, the majority of features is close to a plane, e.g. in the exploration of foreign planets. This property is used in navigation, as in work of Xiao et al. [8] and Zhou et al. [9], and provides reasonable results, if a large number of feature points can be detected. The method

[1,3]Chen Zhu and Patrick Henkel are with the Institute of Communications and Navigation, Faculty of Electrical Engineering and Information Technology, Technische Universität München, 80333 Munich, Germany chen.zhu@tum.de, patrick.henkel@tum.de

[2]Christoph Bamann is with Department of Aerospace Engineering, Faculty of Mechanical Engineering, Technische Universität München, 85748 Garching, Germany christoph.bamann@tum.de

[4]Christoph Günther is with the Institute of Communications and Navigation of the German Aerospace Center, Oberpfaffenhofen and with Technische Universität München, Germany christoph.guenther@dlr.de

This work has been performed in the framework of the project VaMEx (Valles Marineris Explorer) which is partly funded by the Federal Ministry of Economics and Technology administered by DLR Space Agency (FKZ 50NA1212) and the Bavarian Ministry of Economic Affairs, Infrastructure, Transport and Technology administered by IABG GmbH (ZB 20-8-3410.2-12-2012).

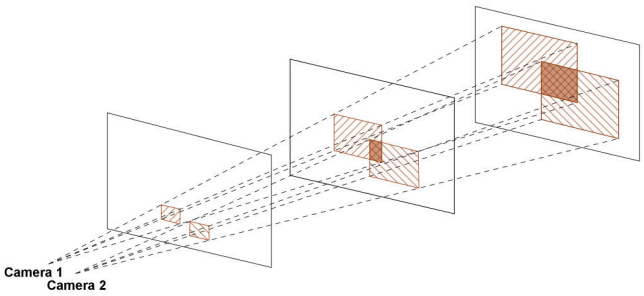

Fig. 1: Depth Impact on Common Field-of-View

proposed in the present paper does not require the features to be on a plane anymore - they can be on a multitude of planes that are automatically determined, and outliers might even be far away from those planes.

Without loss of generality, we consider stereo vision with two cameras which are mounted on two distinct robots in a swarm with time-varying baseline. With known initialization, the attitude changes of the cameras can be tracked by IMUs, so it is acceptable to assume that the orientation of both cameras is known. Similarly the baseline length can be tracked. This allows to triangulate the matched feature points from both views to obtain 3D coordinates. The accuracy of the baseline measurement does not affect the performance of the common FOV algorithm, as shall be seen below.

In the following sections, we use both Cartesian coordinates describing points in Euclidian space $\mathbb{R}$ and homogeneous coordinates describing points in projective space $\mathbb{P}$. The latter have an additional scale coordinate compared with the Cartesian coordinates with same space dimension. In this paper, the 2-norm of a vector is denoted by $\|\cdot\|_2$, and $\langle\cdot,\cdot\rangle$ denotes the inner product of two vectors.

A. Adaptive Fussy Plane Clustering

Assume that n matched feature points with triangulated 3D coordinates $P_1, P_2, ..., P_n \in \mathbb{R}^3$ are the only geometric information about the environment. Then hard clustering defines an initial allocation of the 3D points to K clusters by minimizing the overall distances between data points and centers of the clusters. This allocates every point to a unique cluster. Alternatively, fuzzy clustering defines a level of membership $u_{ik} \in [0,1]$ of point i in cluster k [10]. The center $G_k \in \mathbb{R}^3$ of cluster k is defined by

$$G_k = \frac{\sum_{i=1}^{n}(u_{ik})^m P_i}{\sum_{i=1}^{n}(u_{ik})^m}. \tag{1}$$

and the weights are obtained by minimizing the weighted squared distance from those centers:

$$\sum_{i=1}^{n}\sum_{k=1}^{K}(u_{ik})^m \|P_i - G_k\|_2^2. \tag{2}$$

The exponent m is called fuzzifier. It characterizes the extent of overlap among different clusters. When $m \leq 1$, the optimization becomes equivalent to hard clustering. The cluster centers G_k and membership levels u_{ik} can be calculated iteratively [10]. This defines the initialization of the algorithm.

The points in the individual clusters should be in a plane. In order to determine that, one defines the cluster covariance matrix for the k-th cluster by

$$F_k = \frac{\sum_{i=1}^{n}(P_i - G_k)(u_{ik})^m (P_i - G_k)^T}{\sum_{i=1}^{n}(u_{ik})^m}. \tag{3}$$

The eigenvalues $\lambda_{1k} \leq \lambda_{2k} \leq \lambda_{3k}$ of F_k determine the shape of the cluster prototype. In the present context, the prototypes should be planes, i.e. $\lambda_{1k} \sim 0$. In the case that $\lambda_{1k}/\lambda_{2k}$ is larger than a threshold, the cluster is disregarded.

The allocation of points to clusters was rather arbitrary so far. In the present paper, a large number of cluster prototypes is used. The task is thus to merge clusters that are associated with the same plane. In a first step, we define a metric for measuring the distance from the planes associated with cluster k. Let e_{1k} be the normalized first eigenvector of F_k, then this measure is

$$h_{ik} = |\langle e_{1k}, P_i - G_k\rangle|^2. \tag{4}$$

In order to handle noise and outliers, we introduce a 0-cluster. With these definition the weights are re-computed by minimizing:

$$\sum_{i=1}^{n}\sum_{k=1}^{K}(u_{ik})^m \|h_{ik}\|_2^2 + \sum_{i=1}^{n}(u_{io})^m \delta^2. \tag{5}$$

The rightmost summand makes the clustering process robust. If the distances of point P_i to all the other cluster prototypes are large compared to δ, its membership level u_{io} to the outlier cluster is high, so that it does not affect the calculation of the inlier prototypes.

Next clusters are compared: two clusters k_1 and k_2 are merged whenever the fuzzy inclusion similarity measure $\chi_{k_1k_2}$ between them exceeds a threshold, which is normally chosen to be $1/(K-1)$ [11], i.e. when

$$\chi_{k_1k_2} = \frac{\sum_{i=1}^{n}\min(u_{ik_1}, u_{ik_2})}{\min(\sum_{i=1}^{n}u_{ik_1}, \sum_{i=1}^{n}u_{ik_2})} > \frac{1}{K-1}. \tag{6}$$

The set of new clusters leads to a new computation of the centers G_k according to Equation (1) and of the cluster covariance matrix F_k according to Equation (3) and to another iteration of the algorithms. The iteration ends, after the first iteration without merging.

The eigenvectors e_{2k} and e_{3k} describe the plane γ_k associated with the k-th cluster after merging. This is the basis for computing the common field of view. Let K' be number of such planes.

B. Common Field-of-View Detection

The field of view of a camera is obtained by computing the first intersection of rays with the K' planes. Thus projecting the visible region of one camera on the K' planes and then on the image plane of another camera yields their common FOV.

The visible region of Camera 1 on the detected planes is defined both by the intersections of the planes and the crop due to the limits of the camera's CCD. For each plane i with plane equation $\gamma_i \in \mathbb{P}^3$, we first calculate the ray of the projection of the four points from Camera 1 with that plane:

$$P_{1c}(\rho) = C_1^+ v_{1c} + \rho O_1 \tag{7}$$

where C_1 denotes the camera matrix, $(\cdot)^+$ the pseudoinverse of the matrix, v_{1c} the location of a corner point on image plane, and $O_1 \in \mathbb{P}^3$ is the 3D coordinates of the camera projection center. By solving $\gamma_i^T P_{1c}(\rho) = 0$, we determine the intersection points of the rays with the i-th visible plane. This describes a polygon with four vertices for each plane. Determining the visible part of the surfaces inscribed in these polygons, creates a facetted surface delimited by another polygon. This latter polygon describes the limits of the field of view. The common field of view is obtained by back-projecting the vertices of the latter polygon onto the image plane of Camera 2.

If there are multiple planes, i.e. $K' > 1$, any line l_{ij}, which denotes the intersection line between plane γ_i and γ_j, is an FOV border for those planes. The actual FOV is obtained by combining the FOVs of all planes.

C. Verification of Baseline-scale Invariance

An important property of our common FOV detection approach is that it is invariant under camera baseline scaling. Without loss of generality we simplify the problem by choosing a global frame with Camera 1 at the origin and its orientation aligned with the local frame of that camera. As a result, the camera matrix of Camera 1 is $C_1 = K_1[R_1, O_1] = K_1[I, 0]$, in which K_1 denotes the intrinsic camera matrix, $R_1 \in \mathbb{R}^{3\times3}$ the rotation matrix between camera and the world frame, and $O_1 \in \mathbb{R}^3$ the location of the camera projection center. The second camera is translated by $t \in \mathbb{R}^3$, and rotated by $R \in \mathbb{R}^{3\times3}$. Its camera matrix is $C_2 = K_2[R_2, O_2] = K_2[R, t]$. The vector $t = O_2 - O_1$ is the baseline between the two cameras. With the definition $\Lambda = diag(1,1,1,w)$, scaling of the baseline length by a factor w changes the camera matrix C_2 to $C_2' = K_2[R, wt] = C_2\Lambda$.

Assuming that a 3D point with homogeneous coordinates $P = [X,Y,Z,1]^T \in \mathbb{P}^3$ is projected on the image planes of both cameras with coordinates $v_1 = [x_1, y_1, 1]^T \in \mathbb{P}^2$ and $v_2 = [x_2, y_2, 1]^T \in \mathbb{P}^2$ respectively, implies that $v_1 = C_1 P, v_2 = C_2 P$. Utilizing the cross-product constraints that $v_1 \times (C_1 P) = 0$ and $v_2 \times (C_2 P) = 0$, the unknown 3D coordinates of the point can be triangulated with the 2D coordinates on the two image planes by solving the normal equation

$$AP = \begin{bmatrix} x_1(c_1^3)^T - (c_1^1)^T \\ y_1(c_1^3)^T - (c_1^2)^T \\ x_2(c_2^3)^T - (c_2^1)^T \\ y_2(c_2^3)^T - (c_2^2)^T \end{bmatrix} P = 0 \tag{8}$$

in which $(c_i^j)^T$ denotes the j-th row of the camera matrix C_i.

Replacing the baseline by the scaled baseline, implies replacing the terms in Eqn. (8) by rows $C_1' = C_1\Lambda = C_1, C_2' = C_2\Lambda$ which leads to the equation

$$A'P = A\Lambda P = 0. \tag{9}$$

As a result, if $P = [X,Y,Z,1]^T$ solves Eqn. (8), Eqn. (9) has the solution $P_w = \Lambda^{-1}P = [X,Y,Z,1/w]^T$. Therefore, the impact of scaling the baseline length by w is that the triangulated coordinates of the feature points are scaled by $1/w$. The corresponding Cartesian coordinates have the relation $r_w = [wX, wY, wZ]^T = wr$.

If we compute a plane equation with three non-collinear points from that plane, the Cartesian coordinates of which are triangulated as r_{1w}, r_{2w} and $r_{3w} \in \mathbb{R}^3$ with the same scaled baseline, the plane can be determined by

$$\gamma_w = \begin{bmatrix} (r_{1w} - r_{2w}) \times (r_{2w} - r_{3w}) \\ -r_{3w}^T (r_{1w} \times r_{2w}) \end{bmatrix} \in \mathbb{P}^3. \tag{10}$$

We can normalize the plane as $\gamma_w = [u^T, 1]^T$. From $r_{iw} = wr_i, i = 1,2,3$ it follows that $\gamma_w = \Lambda\gamma$, where γ is the original plane equation calculated with points coordinates r_1, r_2 and r_3.

For point $v_1 = [x_1, y_1, 1]^T$ on the image plane of Camera 1, the back projection results in a ray

$$P(\rho) = C_1^+ v_1 + \rho[0,0,0,1]^T \tag{11}$$

in which ρ is the scalar factor for the parameterized line equation. According to Eqn. (13), the ray intersects plane γ at

$$P_b = \begin{bmatrix} K_1^+ v_1 \\ -u^T K_1^+ v_1 \end{bmatrix} \in \mathbb{P}^3 \tag{12}$$

$$\gamma^T P_b = [u^T, 1][(K_1^+ v_1)^T, \rho]^T = 0. \tag{13}$$

Similarly the ray intersects the plane γ_w at $P_{bw} = [(K_1^+ v_1)^T, -u^T K_1^+ v_1 / w]^T = \Lambda^{-1} P_b$. Here P_b is the original point coordinates while P_{bw} is the corresponding point on the plane calculated with scaled baseline. The projection of P_b on the image plane of Camera 2 yields $v_2 = C_2 P_b$, while P_{bw} is projected to

$$v_{2w} = C_2' P_{bw} = C_2 \Lambda \Lambda^{-1} P_b = v_2. \tag{14}$$

Therefore, the planes induce the same homography between the two cameras for different baseline scales.

The result indicates that our plane-based common field-of-view detection algorithm is invariant under baseline scaling if and only if we back-project the planes using the same baseline parameters that are utilized to triangulate the feature points. Under this assumption, the method is thus robust against baseline length measurement error.

III. Autonomous Robotic Grouping Exploiting Common Field-of-View

In swarm exploration, robots are normally divided into groups to increase the efficiency of exploration and to reduce the communication needs. The groups are evolving over time due to the movement of the autonomous vehicles. The distance of the robots is normally the dominant consideration in grouping. This is motivated by optimizing intra-group

communications and to ease collision avoidance. However, the overlapping FOV of the cameras is essential for swarm VSLAM and is not considered in most cases. Zou and Tan exploit the number of common feature points in their grouping strategy [7]. This is often not a good measure of the overlap of FOVs, however. The grouping metric proposed below is relying on both common FOV and Euclidean distance. It is obtained from the above adaptive common FOV detection method.

A. Similarity Metric based on Field-of-View and Distance

Assume that N robots are to be divided into M groups according to a grouping metric, and that the position of the robots $r_i = [X_i, Y_i, Z_i]^T$ for $i = 1, 2, ..., N$, can be estimated by SLAM, a state-of-the-art choice of a grouping metric d_{ij} is based on distance [12]:

$$d_{ij} = \exp\left(-\frac{\|r_i - r_j\|_2^2}{\sigma^2}\right) \tag{15}$$

with σ being a parameter describing the decay rate of the metric as a function of distance. Close robots are grouped when σ is large. In practice, σ is often chosen as being the standard deviation of the distance among vehicles.

In order to include common FOV in the grouping strategy, we redefine the grouping metric by:

$$s_{ij} = \alpha \cdot (q_i + q_j)/2 + (1 - \alpha) \cdot d_{ij} \tag{16}$$

with $q_i = N_{oi}/N_{pi}$, $q_j = N_{oj}/N_{pj}$ being the ratios of the pixel numbers in the overlapping regions and the total pixel number of image i and j, respectively. The parameter $0 \leq \alpha \leq 1$ allows to tune the relative importance of distance and FOV in the grouping strategy. The grouping metric fulfills $0 \leq s_{ij} \leq 1$ and $s_{ii} = 1$.

B. Autonomous Robots Grouping Using Adaptive Similarity

The grouping metric in Eqn. (16), defines an undirected graph G. Let $r_1, r_2, ..., r_N$ be the positions of N robots, and let them be the vertices of the graph, then connect every vertex-pair $i, j \in \{1, 2, ..., N\}$ by an edge with weight s_{ij}. This is a completely connected graph. The M-grouping problem transforms to a graph cut problem that separates the vertices into M disjoint graphs in which the total weight of the remaining edges is maximized. The graph cut problem is NP-hard. Fortunately, excellent suboptimal solutions can be found by using the graph properties and invariants of graph-based matrices. Spectral clustering algorithms such as the Jordan-Weiss algorithm [12] provide good solutions to the M-grouping problem. Spectral clustering uses the property of graph Laplacian matrices that the number of zero eigenvalues in the Laplacian is the number of connected components in the graph. The Jordan-Weiss algorithm clusters the eigenvectors corresponding to the M largest eigenvalues of the normalized graph Laplacian matrix, which strengthens the cluster property compared with the original data points. In practice, the group number M is usually determined according to the exploration strategy. Nevertheless, if there is a demand to adaptively decide the group number, various criterion can be used to estimate the optimal group number M_{opt}. One option of determining M_{opt} is to maximize the mean intra-group connectivity by

$$M_{opt} = \arg\max_M \frac{1}{M} \sum_{i=1}^{M} \lambda_{2,i,M}, \qquad M = 2, ..., M_{max} \tag{17}$$

where $\lambda_{2,i,M}$ denotes the algebraic connectivity (or Fiedler value) of the i-th cluster, which is the second smallest eigenvalue of the graph Laplacian of a connected graph. A large Fiedler value implies that the connections among the vertices of the graph are strong, while small Fiedler values indicate that many vertices are connected only by links with a small total weight.

IV. Simulation Results

A. Simulations of Common Field-of-View Detection

We tested the common FOV detection algorithm from Section II with several rather different pairs of stereo images. The performance was rather similar in all cases. Thus we choose three typical scenario to illustrate the results. The first scenario is a wall with items on the shelf. The second scenario is the corner of a floor and walls. The third scenario uses a pair of stereo images from the NASA Mars rover Sojourner. For the feature extraction, SURF detector and descriptor [13] are used for all three scenarios. RANSAC (RANdom SAmple Consensus) [14] is implemented to reduce the number of outliers, so that the comparison is fair to the approaches in [5] and [6]. The results are illustrated in Fig. 2, 3, and 4.

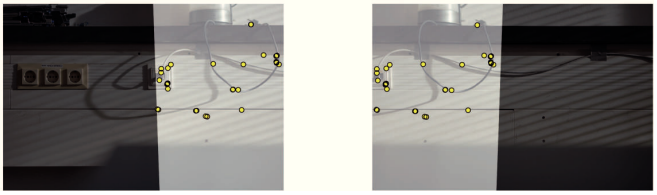

Fig. 2: Common FOV Detection - Wall

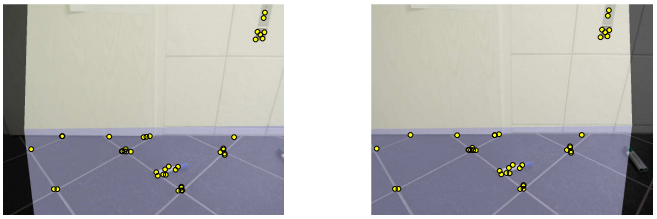

Fig. 3: Common FOV Detection - Floor

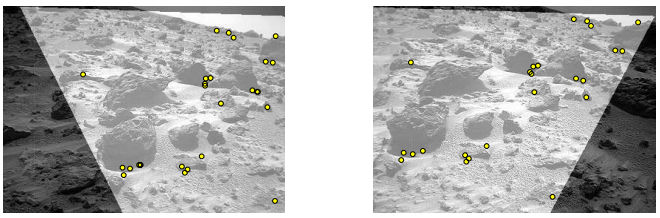

Fig. 4: Common FOV Detection - Mars

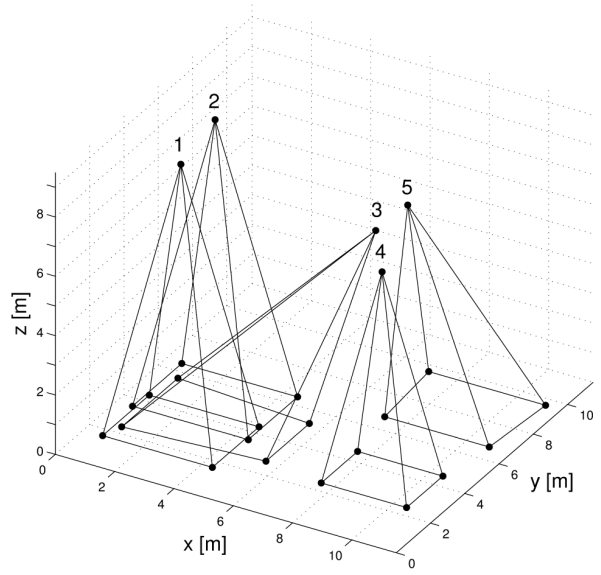

Fig. 5: Perspective of Scenario 1

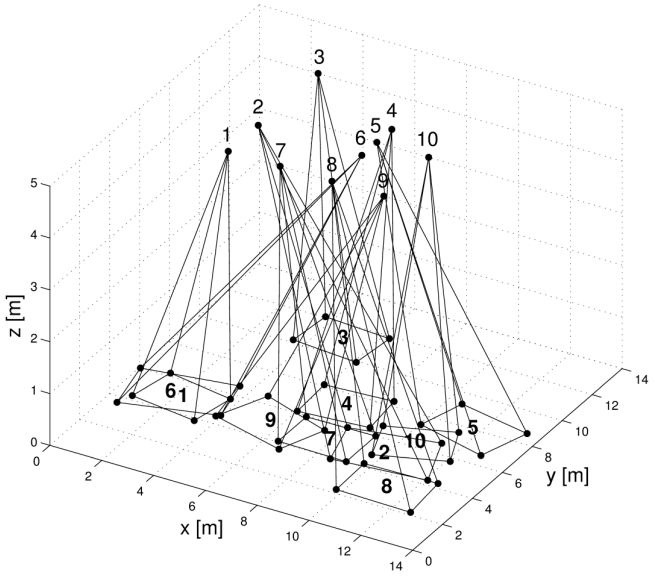

Fig. 6: Perspective of Scenario 2

The highlighted part in the images are the detected common FOV. Yellow dots show the matched feature points. The simulation results indicate that our overlapping detection algorithm performs very well in a wide variety of scenarios.

TABLE I: Common Field-of-View Detection Result

	Detected Plane #	Inlier Points	Overlapped Pixel
Wall	1	30	45.25%
Floor	2	37	91.28%
Mars	1	24	73.50%

Table I compares the values of the overlapping similarity metrics defined by feature point numbers and overlapped pixel numbers. In the "Wall" scenario, 30 feature points are associated with an overlap of 45 percent. On Mars, 24 feature points are associated with an overlap of 70 percent. This shows that the number of matched feature points is not a good measure for the common FOV.

B. Simulations of Autonomous Grouping Algorithm

The autonomous grouping algorithm is simulated in two scenarios. Scenario 1 contains five cameras with different positions and orientations that distributes as Fig. 5.

The robots are grouped using the grouping metric and algorithm introduced in Section III. Assuming that the robots are required to be divided into two groups, different results are obtained when α is varied, see Table II. For small values of α , the distance is the dominant, as expected.

TABLE II: Scenario 1 Grouping Result with $M = 2$

Adaptive Factor	Result Group 1	Result Group 2
$0 \leq \alpha \leq 0.4$	{1,2}	{3,4,5}
$0.5 \leq \alpha \leq 1$	{1,2,3}	{4,5}

Fig. 6 illustrates the cameras distribution in Scenario 2. To make the condition more clear, we fix the height of all the robots in the scenario.

Again we cluster the cameras into two distinct groups according to the adaptive similarity metric for various α values. The grouping result is provided in Table III.

TABLE III: Scenario 2 Grouping Result with $M = 2$

Adaptive Factor	Result Group 1	Result Group 2
$0 \leq \alpha \leq 0.3$	{1,2,3,7}	{4,5,6,8,9,10}
$\alpha = 0.4$	{1,2,3,6,7,8}	{4,5,9,10}
$\alpha = 0.5$	{1,2,6,7,8}	{3,4,5,9,10}
$\alpha = 0.6$	{1,6,7}	{2,3,4,5,8,9,10}
$0.7 \leq \alpha \leq 1$	{1,6}	{2,3,4,5,7,8,9,10}

The increase of α results in a larger impact of common FOV among cameras in the final grouping strategy. To better illustrate it, we plot the average intra-group common FOV and average intra-group distance measure d_{ij} with respect to the incremental adaptive factor α in Fig. 7. It can be concluded from the curve that for very small α, the autonomous vehicles with short distance are clustered together. However, the common FOV inside the groups are extremely small. By setting the adaptive factor to a suitable value, the autonomous grouping algorithm results in a good trade-off between the robots distance and the common FOV of the cameras. The same conclusion can be drawn for other value of M as well.

The simulation results in Fig. 8 illustrate the variation of the autonomous grouping in a dynamic situation. In the scenario, a swarm element moves along the x-axis while the other elements keep static. The number of the clusters is determined adaptively using the method described in Section III-B. In the simulation, α = 0.5, which indicates the distance and the common FOV are treated with equal significance. From Fig. 8 we can observe that in the six snapshots the grouping outcome varies as the swarm element moves, and the clusters are marked with different colors. As the swarm element splits from the red group in (d), 4 clusters are adapted autonomously, and the moving element is clustered into

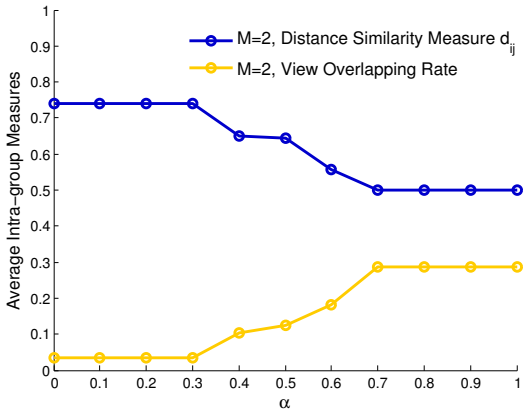

Fig. 7: Average Intra-group Measures of Scenario 2 over α

an independent group. When it further approaches the blue cluster, the moving vehicle is merged into the group. The simulation results indicate reliable grouping performance, and the advantage of using a joint metric of distance and common FOV can be observed. If only the common FOV is considered, the situation in (c), (d) and (e) would be the same, which is intuitively not as reasonable as our results.

V. CONCLUSIONS

In typical environments, feature points can be clustered into planes. These planes are the basis for determining the overlapping Field-of-View (FOV) of pairs of cameras by projection onto and back-projection from those planes. If the same baseline is used for determining the three-dimensional coordinates of the feature points and in the back-projection, the method is invariant with respect to baseline errors. This implies a good level of robustness. The proposed FOV-estimation method outperforms other methods, like those based on counting feature points. It can furthermore be exploited for grouping robots in a swarm in order to achieve a grouping that allows combining images from members of the group to navigate or to explore the environment.

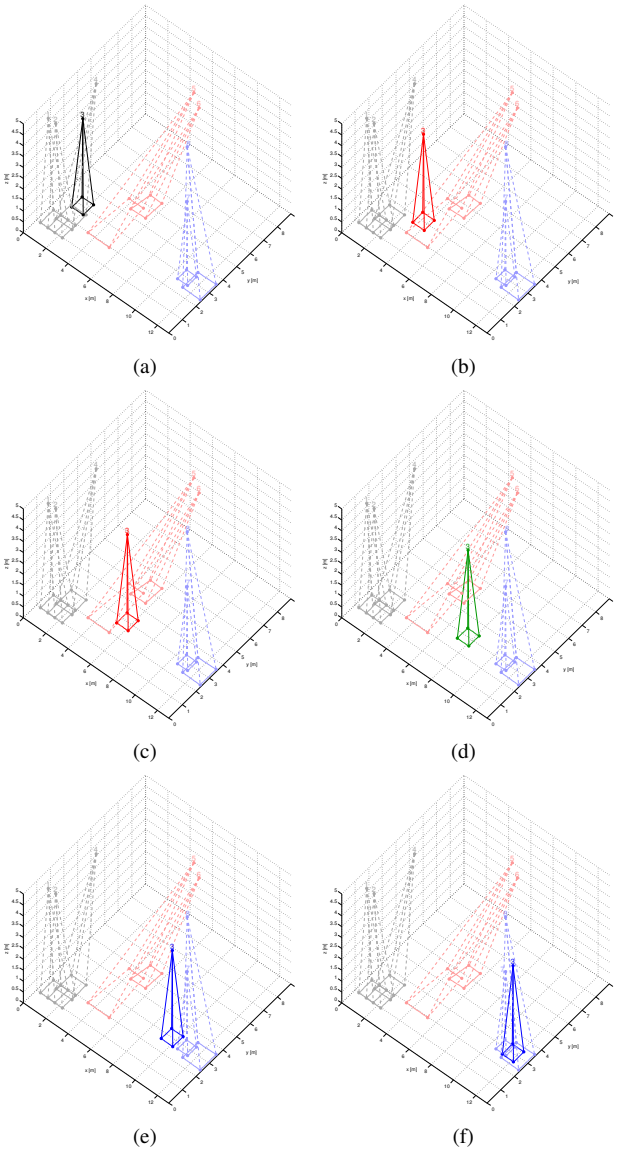

Fig. 8: Splitting and Merging of a Moving Vehicle

REFERENCES

[1] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M. Munich, "The vslam algorithm for robust localization and mapping," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, April, 2005, pp. 24–29.

[2] A. Davison, I. Reid, N. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 1052–1067, June, 2007.

[3] Y. Mohan and S. G. Ponnambalam, "An extensive review of research in swarm robotics," in *Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, Dec. 2009, pp. 140–145.

[4] S. Sand, S. Zhang, M. Mühlegg, G. Falconi, C. Zhu, T. Krüger, and S. Nowak, "Swarm exploration and navigation on Mars," in *Proceedings of International Conference on Localization and GNSS (ICL-GNSS) 2013*, Torino, Italy, June 2013.

[5] N. Snavely, S. M. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *Int. J. Comput. Vision*, vol. 80, no. 2, pp. 189–210, Nov. 2008.

[6] M. Bruckner, F. Bajramovic, and J. Denzler, "Geometric and probabilistic image dissimilarity measures for common field of view detection," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, June, 2009, pp. 2052–2057.

[7] D. Zou and P. Tan, "Coslam: Collaborative visual slam in dynamic environments," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 2, pp. 354–366, Feb. 2013.

[8] J. Xiao, J. Zhang, J. Zhang, H. Zhang, and H. Hildre, "Fast plane detection for slam from noisy range images in both structured and unstructured environments," in *Mechatronics and Automation (ICMA), 2011 International Conference on*, Aug. 2011, pp. 1768–1773.

[9] J. Zhou and B. Li, "Homography-based ground detection for a mobile robot platform using a single camera," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, May, 2006, pp. 4100–4105.

[10] D. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*, vol. 17. IEEE, 1978, pp. 761–766.

[11] U. Kaymak and M. Setnes, "Fuzzy clustering with volume prototypes and adaptive cluster merging," *Fuzzy Systems, IEEE Transactions on*, vol. 10, no. 6, pp. 705–712, Dec 2002.

[12] A. Y. Ng, M. I. Jordan, Y. Weiss, *et al.*, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Computer Vision–ECCV 2006*, pp. 404–417, 2006.

[14] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.