Collaborative Visual SLAM using Compressed Feature Exchange

Dominik Van Opdenbosch, and Eckehard Steinbach

***Abstract*—In the field of robotics, collaborative Simultaneous Localization and Mapping (SLAM) is still a challenging problem. The exploration of unknown large-scale environments benefits from sharing the work among multiple agents possibly equipped with different abilities, such as aerial or ground-based vehicles. In this letter, we specifically address data-efficiency for the exchange of visual information in a collaborative visual SLAM setup. For efficient data exchange, we extend a compression scheme for local binary features by two additional modes providing support for local features with additional depth information and an inter-view coding mode exploiting the spatial relations between views of a stereo camera system. To demonstrate the coding framework, we use a centralized system architecture based on ORB-SLAM2, where energy constrained agents extract local binary features and send a compressed version over a network to a more powerful agent, which is capable of running several visual SLAM instances in parallel. We exploit the information from other agents by detecting overlap between already mapped areas and subsequent merging of the maps. Henceforth, the participants contribute to a joint representation and benefit from shared map information. We show a reduction in terms of data-rate by 70.8% using the feature compression and a reduction in absolute trajectory error by 53.7% using the collaborative mapping strategy with three agents on the well-known KITTI dataset. For the benefit of the community, we provide a public version of the source code.**

***Index Terms*—Multi-Robot Systems, SLAM, Localization, Mapping, Visual-Based Navigation.**

I. INTRODUCTION

MANY robotic tasks are only feasible with profound knowledge about the environment, the current position of the robot, and the location of possible obstacles. With the advent of affordable, versatile and compact visual sensors, visual SLAM became a very practical approach to solving the SLAM problem. While small-scale SLAM systems are available today, the research focus shifts towards large-scale mapping. Using not only a single, but multiple agents in such scenarios reduces the required mapping time and adds fault tolerance in case of agent failure. Therefore, collaborative visual SLAM has recently attracted some attention in the scientific community. In this letter, we focus on a scenario where a small team collaboratively explores the surroundings.

Manuscript received: July, 24, 2018; Revised September, 27, 2018; Accepted October, 19, 2018.

This paper was recommended for publication by Editor Nak Young Chong upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the space agency of the German Aerospace Center with funds from the Federal Ministry of Economics and Technology on the basis of a resolution of the German Bundestag (FKZ: 50NA1515).

D. Van Opdenbosch and E. Steinbach are with the Chair of Media Technology, Technical University of Munich, Munich 80333, Germany. dominik.van-opdenbosch@tum.de

Digital Object Identifier (DOI): see top of this page.

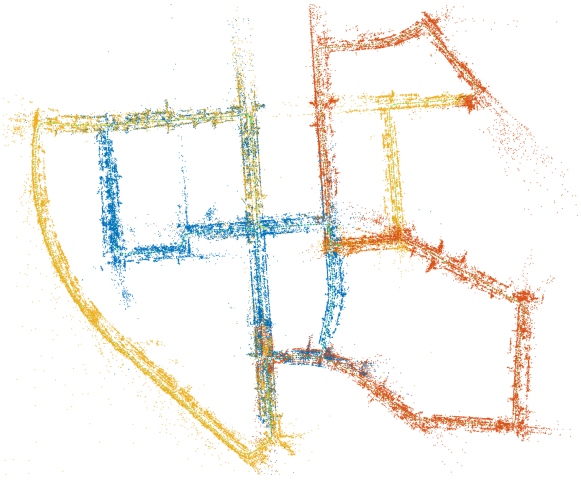


Fig. 1. Globally consistent map after merging individual maps obtained from three clients operating on the KITTI 00 sequence. The color indicates which agent created the map points.

The requirements in terms of energy efficiency are dependent on the agent type. While for Micro Aerial Vehicles (MAV) energy efficiency is a key issue for long-term operation, rovers are usually not as restricted regarding energy consumption and can perform computationally more complex tasks by carrying additional processing capabilities and the required batteries. In order to exploit this difference, we use a centralized system architecture where only the visual feature extraction runs at the energy constrained clients and compressed visual cues necessary to run metric scale visual SLAM are transmitted to the central server carried by a powerful agent. This server runs individual visual SLAM instances for each client, performs the map building, and if necessary transmits the position and map information back to the respective client. Additionally, the server is capable of detecting overlaps between the visual SLAM maps and performs map merging. After a merge occurred, the attached agents will contribute to a common map, as shown in Figure 1. As an example, this global map can later be used for the central orchestration of the exploration team. To this end, this work contains the following contributions:

1) We extended our existing coding framework for local binary features [1] to include additional depth information from RGB-D cameras or stereo feature matching to allow metric scale visual SLAM.
2) We complete the coding framework with a stereo feature coding mode capable of transmitting the features from both views by exploiting spatial correlations.
3) To demonstrate the coding in a collaborative scenario, we implemented a centralized visual SLAM system.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2878920, IEEE Robotics and Automation Letters

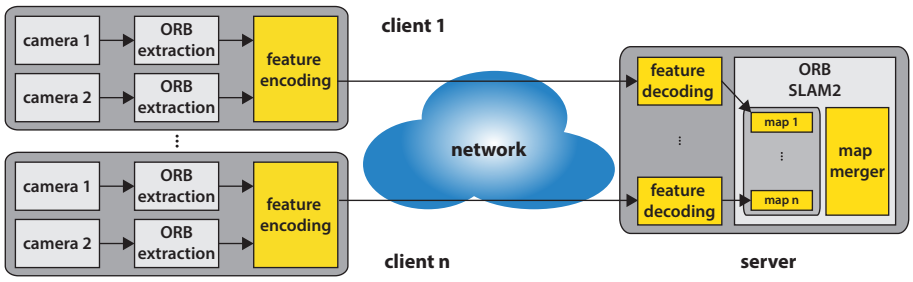


Fig. 2. System overview: Client agents run only visual feature extraction and feature encoding. The server decodes the feature streams and creates the individual maps for all clients. A map merging module is triggered on creation of new keyframes and merges overlapping maps into a unified representation.

The rest of the letter is organized as follows: In Section II, we discuss the related work for visual SLAM, collaborative visual SLAM, and local feature compression. In Section III, we introduce our system architecture. Section IV details the coding framework including the proposed depth and stereo coding. Section V introduces the collaborative mapping approach. We demonstrate the effectiveness of the system in Section VI and draw a conclusion in Section VII.

II. RELATED WORK

A. Visual SLAM

Most visual SLAM systems can be categorized according to two key properties [2]. They can be either direct or indirect and dense or sparse methods. While indirect methods use an intermediate representation such as local features or templates based on image patches, the direct methods operate on the raw pixel intensities. Dense methods try to reconstruct a full 3D model of the environment based on all pixels, whereas sparse methods only use a subset of image points to create 3D points. Representatives of both direct and dense methods are the Dense Tracking and Mapping (DTAM) [3] and Large-Scale Direct SLAM (LSD SLAM) [4]. Both try to minimize a photometric error term between frames based on the raw pixel intensities to estimate the motion, which is computationally demanding. Therefore, some methods combine the direct with the sparse approach, such as Direct Sparse Odometry (DSO) [2]. It only provides visual odometry without loop closing, which is crucial to correct drift in large-scale mapping scenarios. For loop closing, it has to rely on feature-based methods [5]. Other approaches, like Fast Semi-Direct Monocular Visual Odometry (SVO) [6] use direct methods for motion estimation and indirect approaches for building a map. Direct methods usually require the raw pixel intensities to be available, which requires a lot of data exchange in collaborative scenarios. Approaches that are both indirect and sparse have attracted attention for collaborative applications due to their image abstraction and sparse map representation. Examples for indirect and sparse systems are MonoSLAM [7] and PTAM [8]. A recent representative is ORB-SLAM2 [9], [10], which is a comprehensive SLAM framework capable of real-time parallel tracking, mapping, loop closing and re-localization.

B. Collaborative Visual SLAM

With accurate real-time capable visual SLAM systems at hand, collaborative mapping approaches have received increasing interest. Zou et al. [11] proposed a system named CoSLAM using multiple cameras simultaneously. They group cameras according to their view overlap and propose to collaboratively map the environment using both intra- and inter-camera mapping. Forster et al. [12] proposed a monocular SLAM system for joint mapping using MAVs. Riazuelo et al. [13] proposed a cloud-based approach for cooperative tracking and mapping. They outsource the costly optimization problem to more powerful processing nodes in the cloud and keep only a lightweight tracking on the local device. Schmuck et al. [14] proposed a collaborative SLAM system for multiple MAVs based on ORB-SLAM2. In their concept, each MAV runs a lightweight visual SLAM system and the information is collected at a central server, where the computationally demanding tasks, such as map optimization, are carried out. Later, the authors extended their system to a visual-inertial collaborative SLAM system [15]. Regarding the map exchange, approaches for map synchronization have been proposed in [16], [17]. In order to restrict the unbounded growth of maps and reduce the data footprint when exchanging static map information, several works addressed the issue of map sparsification keeping only a limited number of useful map points [18], [19], [20] or optimizing for a target map size in terms of the number of required bits [21]. Fully decentralized systems have been proposed by Cieslewski et al. [22] based on previous work on distributed overlap detection [23], [24]. Their system allows exchanging quantized local features in form of visual word indices, as proposed by [25]. However, most of the distributed systems ignore the aspect of efficient on-the-fly compression for exchanging the visual information.

C. Feature Compression

Addressing feature compression, Baroffio et al. introduced the concepts of hybrid video coding to both real-valued [26] and binary feature descriptors [27]. They proposed different coding modes such as intra- and inter-frame coding to exploit both the dependencies among feature descriptor elements and the temporal correlation between successive frames. Additionally, inter-view coding has been introduced by Bondi et al. [28]. Redondi et al. [29] provided an analysis of visual sensor network performance using the principles of inter-view coding. In previous work, we addressed the joint coding of local and global image features by exploiting the dependencies between the descriptors and their corresponding Bag-of-Words representation [30]. Some of the aforementioned paradigms

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2878920, IEEE Robotics and Automation Letters

form the basis of the MPEG CDVS (Compact Descriptors for Visual Search) standard [31], which aims at efficient descriptors for visual search tasks and image retrieval, but has also been used in the context of visual SLAM [32]. More recently, some effort was made towards efficient descriptors for video analysis for the MPEG CDVA (Compact Descriptors for Video Analysis) standard [33] including the possibility to exploit temporal dependencies and deep-learned feature descriptors. Visual descriptor compression using product quantization has been applied to visual maps in [34]. An overview of different compact feature representations is given in [35].

III. SYSTEM OVERVIEW

Our system is based on our previous work [1], where we combined the concept of coding binary features with the monocular version of ORB-SLAM2. In this letter, we propose three extensions to this approach to obtain a metric scale collaborative visual SLAM system based on stereo information, as depicted in Figure 2. First, we use a centralized architecture, where clients take images, extract and encode ORB features [36], and send them over a network to a server running multiple feature decoding instances, which handle all incoming data streams. After decoding, the features are forwarded to their corresponding visual SLAM instance. Each visual SLAM instance consists of a tracking, mapping, and loop closing module operating on the map assigned to this agent. In order to exploit the collaborative mapping aspect, we implemented a map merging module. Similar to the map fusion approach by Schmuck et al. [14], we constantly check new keyframes inserted in any map for correspondences in any other map in a separate background thread. In case overlap is detected, the maps are fused and the attached agents henceforth contribute to a consistent common map.

For SLAM using only visual cues, a depth sensor or a calibrated multi-camera setup is essential to provide metric scale information. Hence, the next contribution is an extension of the monocular feature coding to include depth values obtained either directly from an RGB-D camera or by stereo feature matching. Instead of sending the floating point depth representation, we employ non-uniform scalar quantization tailored to typical depth values used in visual SLAM.

For the last contribution, we add a stereo-view coding scheme exploiting the spatial correlation between both views in a stereo camera setup. In order to efficiently code the visual information, we use the basic concept known from coding features extracted from general visual sensor networks [28], where we transmit only differences between matched features across both views. With the proposed contributions, we address the need for efficient compression in collaborative visual SLAM systems.

IV. FEATURE CODING

A. Coding Framework

We will briefly recapitulate the basic binary feature coding scheme as introduced in our previous work [1] and then detail the extension to depth and stereo coding. The input of the used visual SLAM system are ORB features, which consist of two

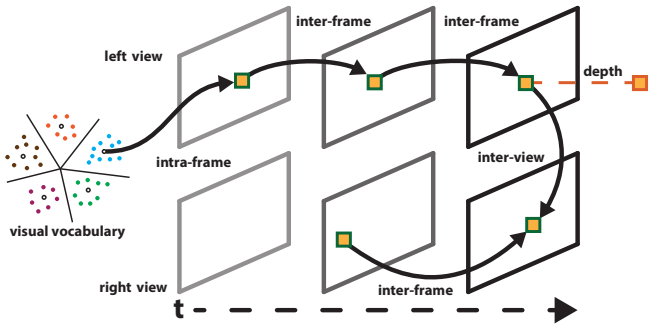


Fig. 3. Prediction modes exploiting either a visual vocabulary (intra-frame), temporal (inter-frame) or spatial correlations (inter-view) between local features. Alternatively, quantized depth information can be transmitted.

parts: The first part describes the location, orientation, and scale where the local feature has been detected, commonly referred to as keypoint. We follow our notation in [1] and denote the set of keypoint properties for feature i from image n as $\mathbf{k}_{n,i} = [x, y, \sigma, \theta]$, where x and y are the keypoint positions given in pixel coordinates, σ denotes the scale level and θ the keypoint orientation. The second part of a local feature is the descriptor, which is the outcome of a set of pairwise pixel tests contained in a binary string $\mathbf{d} \in [0, 1]^D$ with length $D = 256$.

In our previous work [30], we introduced an intra-frame coding mode, which exploits the dependencies between the descriptors and their corresponding Bag-of-Words representation. We use a visual vocabulary as shared knowledge, calculate the difference vector between the descriptor and the closest visual word and transmit only this residual vector containing the binary differences alongside the corresponding visual word index. The difference vector contains mostly zeros, which, in combination with an entropy coding approach (in our case arithmetic coding), leads to a reduction in bitrate. Besides the advantage of exploiting the visual vocabulary to minimize the entropy of the transmitted data, ORB-SLAM2 can directly reuse the visual words in a Bag-of-Words representation for efficient feature matching, loop closure detection, and re-localization. In order to exploit temporal correlation, we added an inter-frame coding mode [1] similar to the approach proposed by Baroffio et al. [27], where only the differences of the keypoint and descriptors with respect to a feature contained in the last frame alongside an identifier have to be transmitted. Additionally, we added a skip mode which is able to copy reference features from previous frames without further coding, thus reducing the bitrate when no motion is present in the video. For details, we refer the reader to [1].

Starting from these coding modes, we complete the framework by adding a coding mode for additional depth information and a stereo feature coding mode. Throughout the letter, we denote the coding modes as $m \in \{I, P, S, M\}$, where I denotes intra-frame coding, P predictive inter-frame coding, S skip mode, and M the new multi-view stereo coding mode. We use +D to indicate the presence of additional depth data. The different concepts are illustrated in Figure 3. For the mode decision, we calculate the rate for all coding modes for each feature (I, P, S for the left view and additionally M for the right view) and select the mode with the lowest bitrate.

B. Depth Coding

The depth coding can be used to efficiently add depth information to each feature of the left view. The depth is acquired either directly by an RGB-D camera or by stereo feature matching. For the latter, we assume a pre-calibrated camera system with known camera intrinsic and extrinsic calibration. We identify feature matches located on the same scale by searching along the epipolar line. To account for imperfect camera calibration, we allow a deviation of two pixels from the epipolar line in the scale-space representation. Instead of directly signaling the floating point representation requiring 32 bits per depth value, we propose to use a *non-uniform quantization* scheme trained on sample trajectories to minimize the reconstruction error. Due to occlusion, not every feature is accompanied by a depth value, so for every feature we first send a single bit indicating if a depth value is following. If depth is available, we signal the reconstruction level using a fixed number of N_D bits. In this scheme, the depth estimation is performed at the client side and all features from the right view can be omitted.

C. Inter-View Coding

More general, when using a visual SLAM application, that makes use of the visual information of both views, it is necessary to transmit all features from both cameras. To this end, we propose to exploit the *spatial correlation* between both views by finding feature correspondences and send only the differences between the descriptors alongside an identifier for the reference feature and the missing information to reconstruct the keypoint information. Due to occlusions, this is not always possible and visual features close to the camera system might result in considerably different feature descriptors, as the viewing angle differs substantially. We designed our system such that for each feature in the right view the coding modes introduced for single view feature coding can be chosen as well. To establish correspondences, we use the same stereo matching as for the depth value estimation. The coding costs $R^M_{n,i}$ in bits for inter-view coding for feature i from image n consists of the individual costs for coding the reference feature index $R^{M,ref}_{n,i}$, the costs for the residual vector containing the differences between the matched descriptors $R^{M,des}_{n,i}$ and the cost for the keypoint properties $R^{M,kpt}_{n,i}$ given as

$$R^M_{n,i} = R^{M,ref}_{n,i} + R^{\mathrm{M},des}_{n,i}(j) + R^{M,kpt}_{n,i}, \tag{1}$$

where j denotes the reference feature. After we have identified the set of candidate features from the left view near the epipolar line, we test for the feature j^* that has the minimum Hamming distance H between the current descriptor $\mathbf{d}_{n,i}$ and the reference descriptor $\mathbf{d}^M_j$

$$j^* = \arg\min_j (H(\mathbf{d}_{n,i}, \mathbf{d}^M_j)). \tag{2}$$

1) Reference Coding:
Assuming uniform probability, the entropy for coding the reference feature is

$$R^{M,ref}_{n,i} = log_2(N_{l,n}), \tag{3}$$

where $N_{l,n}$ denotes the number of features in the left view.

2) Descriptor Coding:
For coding the residual, we calculate the difference vector between the current and the reference descriptor in terms of the binary XOR operation $\mathbf{r}^M_{n,i} = \mathbf{d}_{n,i} \oplus \mathbf{d}^M_{j^*}$. With $h^M_{n,i}(j^*)$ denoting the Hamming distance between the current and the reference feature given by

$$h^M_{n,i}(j^*) = H(\mathbf{d}_{n,i}, \mathbf{d}^M_{j^*}), \tag{4}$$

the minimum number of bits can be calculated using the binary entropy function as follows

$$\begin{aligned} R^{M,des}_{n,i}(j^*) = &-(D - h^M_{n,i}(j^*)) \cdot log_2(p^M_0) \\ &- h^M_{n,i}(j^*) \cdot log_2(1 - p^M_0), \end{aligned} \tag{5}$$

where D is the length of the descriptor and p^M_0 is the probability of any entry of the residual vector being zero for the inter-view coding mode. This lower bound is approached using arithmetic coding applied to each residual vector entry. In this scheme, the selected feature j^* is not only the best stereo match, but also results in the minimum coding costs.

3) Keypoint Coding:
For the keypoints, we transmit the quantized orientation and keypoint position. We restrict feature matches to the same scale $\sigma_{n,i}$ allowing to directly reuse the scale information of the reference feature. We encode the x position similar to the intra coding approach [1], where we scale the keypoint position into the scale-space level, where the feature has been extracted. In this representation, the keypoint coordinates are located at integer positions, thus allowing lossless keypoint position coding. For the y-coordinate, we just transmit the difference with respect to the position of the reference keypoint in the common scale-space. We allow a deviation of $\pm$ 2 pixels from the epipolar line in the corresponding scale-space representation requiring $\log_2(5)$ bits for signaling. The orientation is quantized into $N_{\hat{\theta}}$ bins. Assuming uniform distributions, the total keypoint costs can be written as

$$R^{M,kpt}_{n,i} = \log_2(width(\sigma_{n,i})) + \log_2(5) + \log_2(N_{\hat{\theta}}), \tag{6}$$

where $width(\sigma_{n,i})$ denotes the width of the image at the respective scale level $\sigma_{n,i}$. The quantized orientation and x-component could also be coded differentially and individual probabilities could be estimated, but this provides only marginal gain compared to the computational overhead.

V. COLLABORATIVE MAPPING

A. Map Merging

Map merging is facilitated by adapting the existing techniques for loop closing of ORB-SLAM2 to work across multiple maps. First, possible loop candidates are detected by evaluating the visual similarity between the current keyframe and the possible target keyframes contained in all available maps using the Bag-of-Words representation. After obtaining an initial set of candidates, a similarity transformation between the map points contained in the matched keyframes is calculated using the method of Horn [37]. This step serves as geometric verification of the matches to avoid false map mergers. After a successful validation, the two maps are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2878920, IEEE Robotics and Automation Letters

TABLE I
COMPARISON OF THE PERFORMANCE OF DIFFERENT DEPTH QUANTIZERS ON KITTI 00 SEQUENCE IN TERMS OF ABSOLUTE TRAJECTORY ERROR OBTAINED FROM FIVE RUNS.

Quantizer N_D	KITTI 00 [m] min / max / median
4 bit	5.41 / 5.77 / 5.52
5 bit	1.89 / 2.01 / 1.97
6 bit	1.38 / 1.47 / 1.41
7 bit	1.27 / 1.32 / 1.29
8 bit	1.23 / 1.28 / 1.24
32 bit (no quant.)	1.21 / 1.26 / 1.23

TABLE II
MEDIAN TIMINGS AND BITS PER FEATURE FOR MONOCULAR + DEPTH ENCODING AND DECODING MEASURED ON THE KITTI 00 SEQUENCE WITH DIFFERENT MODE CONFIGURATIONS.

Intel Core i7-7700	I+D	I+P+S+D
encoding	11.1 ms	14.5 ms
decoding	12.4 ms	12.7 ms
TX2	I+D	I+P+S+D
encoding	26.5 ms	38.6 ms
decoding	22.9 ms	25.1 ms
bits/feature	229.0	210.6
# features	2k	2k

merged by adding all keyframes and map points from the source map to the target map. The next step is to align the keyframes and map points at the overlap using the estimated similarity transformation. This allows us to identify more duplicate map points and establish connections between map points from the source map and keyframes in the target map and vice versa. After this step, an optimization of the Essential Graph, which is a spanning tree between the keyframes is performed. To this end, we connect the Essential Graphs of both maps. Subsequently, a global bundle adjustment is triggered asynchronously in the background to optimize the common map.

B. Joint Mapping

During map merging, the tracking and the mapping thread of the source map are attached to the target map. The loop closing thread of the source map is stopped. Henceforth, the loop closing thread of the target map is solely responsible to detect loops within the joint map. In order to avoid concurrent access and to keep consistency, we use a shared locking scheme, where only one mapping thread is allowed to add new map points to a particular keyframe. In addition, only one thread can update the keyframe and map point position at a time.

VI. EXPERIMENTAL EVALUATION

For the sake of consistency and to prove generalization, we used the same parameters as in our monocular evaluation [1] and trained the probabilities p_0^M for stereo coding on the same training data as for previous the version, namely the EuRoC dataset [38] Machine Hall sequences. It consists of images collected on-board a Micro Aerial Vehicle with a resolution of 752x480 pixels at 20 Hz in an industrial environment. We evaluated our scheme on the well-known KITTI dataset [39]. It features different sequences collected by a car in an urban scenario with a calibrated stereo camera setup. The undistorted images have a resolution of 1241x376 pixels captured at 10 Hz. The KITTI dataset poses a challenging scenario for both the feature coding due to the large distances between the frames, as well as the mapping aspect due to the spatial extent of the covered area. To obtain the quantization characteristics for the depth value, we used both the Machine Hall and KITTI sequences to train the codebook for the depth values to account for the different depth ranges in indoor and outdoor scenarios.

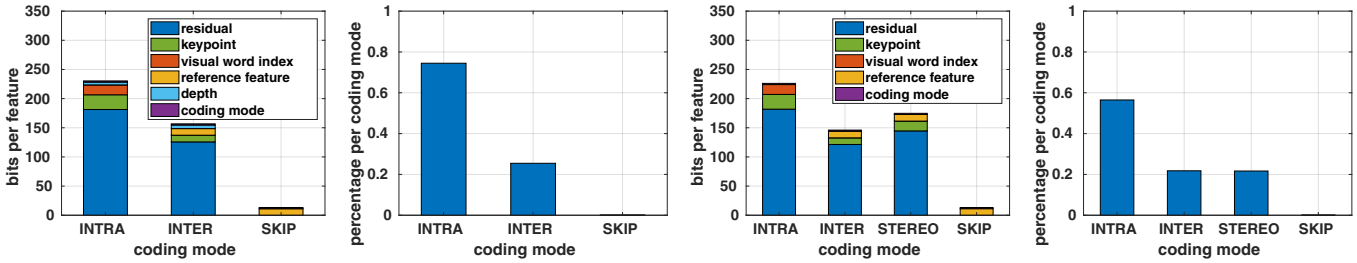
A. Feature Coding

First, we provide some insight on the depth value coding. We present the mapping results in terms of *absolute trajectory error* (ATE) [40] depending on the number of bits for quantizing the depth values for the KITTI 00 sequence in Table I using the default setting of ORB-SLAM2. With $N_D = 8$ bits per depth value and with an additional bit signaling whether a depth value was estimated, we achieved a performance on par with the original depth values. As ORB-SLAM2 uses the depth information to reproduce the original stereo coordinates [10], the result of unquantized depth and stereo coding is similar.

In our second experiment, we present the results of the proposed feature compression scheme for the KITTI 00 sequence in Figure 4. For intra coding, we spend around 225.4 bits for both left and right view without depth information. Inter coding is restricted to use only the past frame for prediction and requires 150.3 bits and 145.8 bits for the left and right view, respectively. Adding more reference frames would reduce the bitrate at the drawback of increased complexity. Depth values for the left view could be estimated for about 47.9% of the features using intra coding and 64.5% of the features using inter mode. On average, this adds 4.8 bits to the intra coding and 6.2 bits to the inter coding mode in the left view. Intuitively, inter coded features are considered to be more stable and not located at the image boundary so that they could be tracked over multiple frames or observed from neighboring views, resulting in the different percentages. The skip mode, which encodes only the reference feature index in the past frame takes 13.3 bits, but is only used for less than 1% of the features in this sequence. All coding modes include an overhead of two bits for signaling the mode. The proposed stereo coding mode uses 11.3 bits for the reference, 144.6 bits for the residual information and 16.9 bits for the keypoint differences. Including the signaling costs, a stereo coded feature takes on average about 174.8 bits. We also show the fraction of features coded with a particular coding mode. Due to the large inter-frame distance, many features are coded using the intra coding mode. The fraction of inter coded features for a high frame rate dataset like EuRoC is significantly higher thus making the coding more efficient.

We measured the coding time on two different systems. First, we show the median timing results on an Intel Core i7-7700 with 3.6 GHz measured on the KITTI 00 sequence for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2878920, IEEE Robotics and Automation Letters

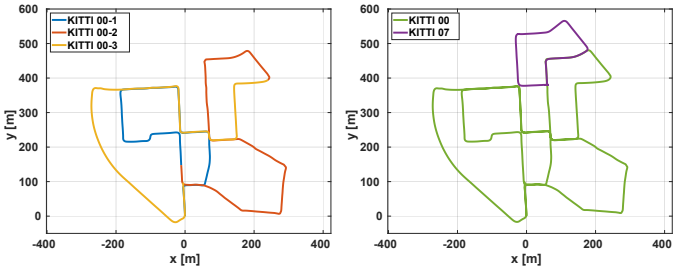


(a) Bits per mode for the left view. (b) Mode decision for the left view. (c) Bits per mode for the right view. (d) Mode decision for the right view.

Fig. 4. Comparison of the number of bits required for intra coding, temporal prediction, skip mode, and inter-view stereo prediction as mean values per feature over the KITTI 00 sequence. The left view includes the bits required for depth value coding. We show the fraction of features using a particular coding mode for the left (b) and the right (d) view, respectively.

TABLE III
MEDIAN TIMINGS AND MEAN BITS PER FEATURE FOR STEREO ENCODING AND DECODING MEASURED ON THE KITTI 00 SEQUENCE WITH DIFFERENT MODE CONFIGURATIONS.

Intel Core i7-7700	I	I+M	I+P+S+M
ORB	20.9 ms		
encoding	17.3 ms	18.7 ms	25.9 ms
decoding	20.5 ms	20.7 ms	21.3 ms
TX 2	**I**	**I+M**	**I+P+S+M**
ORB	67.3 ms		
encoding	52.9 ms	59.5 ms	86.0 ms
decoding	48.4 ms	48.2 ms	49.5 ms
bits/feature	224.3	216.5	201.3
# features	2x2k	2x2k	2x2k



(a) KITTI 00 and 07 sequence. (b) KITTI 00 split into three parts.

Fig. 5. Overview of the two sequences from the KITTI dataset used for the experimental evaluation shown in a common coordinate system.

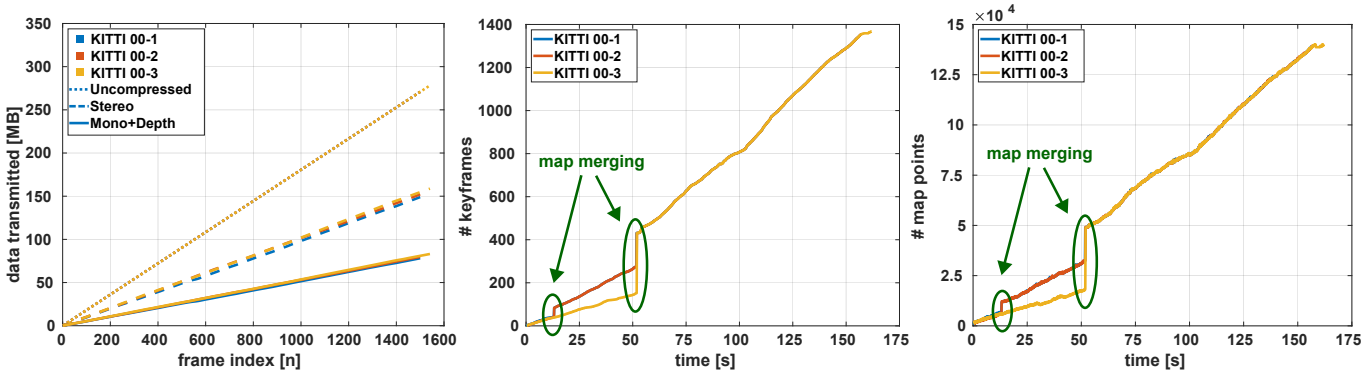
the monocular + depth coding for two coding profiles in Table II. Although, in this scheme we transmit only the 2k features from the left view, the depth estimation needs the features extracted from both views. In the first column, we allow only intra mode plus depth information (I+D). This mode does not use any temporal correlations in the video stream. This allows us to start decoding at any frame without any prior knowledge about previous frames, also known as *random access*. Moreover, this is the fastest coding mode and can be used in *low-delay* scenarios at the drawback of increased mean bitrate. Next, temporal prediction and feature skipping are allowed (I+P+S+D) resulting in an increase in coding time but also a reduction in the mean number of bits per feature. In Table III, we evaluated the stereo coding mode for three different coding profiles. In the first column, we allow only intra mode (I) for both views. Next, we add our proposed inter-view coding mode (I+M) which results in a slight increase in encoding time but at the advantage of a decreased number of bits per feature. Next, we add temporal prediction and the skip mode (I+P+S+M), which adds significant processing time, but also provides a considerable reduction in bitrate.

We also performed several tests on an NVIDIA Jetson TX2 in MAX-P ARM mode using 7.2 Watt at 2.0 GHz included in Table II and III. We did not perform any specific optimization for the embedded platform. It can be seen that the ORB feature extraction running in parallel on two images poses a bottleneck on running ORB-SLAM2 or our system on the embedded platform. However, the system is capable of encoding 2k features including depth estimation in about 26.5 ms. For many computer vision applications or visual SLAM scenarios, fewer features might be sufficient thus lowering the encoding time. In addition, optimized local binary features for embedded devices are available [41], but beyond the scope of this letter.

B. Collaborative Mapping

In order to demonstrate the collaborative mapping aspect, we performed an experiment with two agents operating on the KITTI 00 and 07 sequences, as shown in Figure 5a. We simultaneously ran both sequences and evaluated the result in terms of feature coding and SLAM precision. The following experiments were conducted on a virtual machine running in the cloud with 16 vCPUs based on an Intel Xeon Platinum 8124M at 3.00 GHz. In order to obtain a baseline for the accuracy achievable with our setup, we first conducted experiments for both sequences with map merging deactivated, which is denoted as standalone. Afterwards, we activated the map merging module allowing to incorporate map information from both agents. We show the results in terms of ATE evaluated on each trajectory individually obtained from five individual runs in Table IV. The results show an improvement by 5.7% for KITTI 00 and 16.9% for the KITTI 07 sequence. At the end of the mapping process, a fully connected map covering both sequences is obtained.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2018.2878920, IEEE Robotics and Automation Letters



(a) Cumulative data rate. (b) Number of keyframes. (c) Number of map points.

Fig. 6. Analysis of the cumulative data rate measured at the encoder per agent, where stereo uses the I+P+S+M modes and mono+depth the I+P+S+D modes. Next to it the evolution of keyframes and map points of the SLAM system using three agents on the KITTI 00 sequence.

TABLE IV
COMPARISON OF THE PERFORMANCE ON KITTI 00 AND 07 SEQUENCES WHEN USING THE PROPOSED COLLABORATIVE APPROACH IN TERMS OF ABSOLUTE TRAJECTORY ERROR OBTAINED FROM FIVE RUNS.

Sequence	Standalone [m] min / max / median	Collaborative [m] min / max / median
KITTI 00	1.21 / 1.26 / 1.23	**1.10 / 1.26 / 1.16**
KITTI 07	0.60 / 0.70 / 0.65	**0.48 / 0.66 / 0.54**

TABLE V
COMPARISON OF THE PERFORMANCE ON KITTI 00 SEQUENCE WHEN USING THE PROPOSED COLLABORATIVE APPROACH IN TERMS OF ABSOLUTE TRAJECTORY ERROR OBTAINED FROM FIVE RUNS.

Sequence	Standalone [m] min / max / median	Collaborative [m] min / max / median
KITTI 00-1	**0.99 / 1.09 / 1.08**	1.17 / 1.26 / 1.21
KITTI 00-2	1.44 / 1.52 / 1.49	**1.07 / 1.37 / 1.14**
KITTI 00-3	2.17 / 3.45 / 3.11	**1.31 / 2.18 / 1.44**

In our next experiment, we used three agents jointly on the KITTI 00 sequence. To this end, we divided the sequence into three disjunct parts, as shown in Figure 5b. We simultaneously started all sequences and at some point, significant overlap between the maps is detected and the maps are successively aggregated into a single map. The results in terms of ATE obtained from five individual runs are given in Table V. The first agent achieves a slightly smaller error when running in standalone mode, whereas the results clearly improve for the other sequences when using the map merging functionality. For example, the error of the third agent can be reduced from 3.11 m to 1.44 m which is a reduction by 53.7%. The improvement comes partially from re-using mapped areas and also from additional loops detected within the merged maps allowing to correct drift. Figure 6a shows the amount of data exchanged with each client agent and the server. We compare the cumulative data rate required to exchange the features using uncompressed transmission, stereo feature coding and depth coding for all agents. The achievable bitrate reduction is about 44.1% using stereo coding and 70.8% using mono+depth by assuming 360 bits per uncompressed feature, which is the sum of 256 bits for the descriptor, 3x32 bits for signaling x, y and the feature orientation and additional 8 bits for the scale information. We show the evolution of the maps over time in terms of keyframes in Figure 6b and in terms of map points in Figure 6c. We highlight the occurrences of map merging where the existing keyframes and map points are aggregated into a joint representation. Using this approach, we reduced the time to build a complete and consistent map of the KITTI 00 sequence, as shown in Figure 1, to a third of the original sequence length by using three agents simultaneously. Additionally, we measured the median time required for tracking individual frames with 28.8 ms for the agent operating on KITTI 00-3. This corresponds to the timings reported by the ORB-SLAM2 authors [10] without the ORB feature extraction which is outsourced to the client. In our centralized setup, the number of clients is limited by the computational power of the central server. For larger teams, extending the system to support multiple servers that are capable of exchanging information is required.

VII. CONCLUSIONS

In this letter, we present a comprehensive framework for collaborative visual SLAM for a small team. To this end, we extend our previous binary feature coding framework with a depth and an inter-view coding method. This framework can be used standalone in many application scenarios and is not limited to visual SLAM. In addition, we implemented a collaborative mapping scheme based on ORB-SLAM2 where multiple SLAM maps are built in parallel and can be merged when overlap between the maps is detected. We combine both approaches into a system architecture, where the computationally demanding visual SLAM system is running on a central processing node and only the visual cues are extracted and compressed at the client agents. We evaluated our approach in terms of coding efficiency, timing and absolute trajectory error on the KITTI dataset showing a substantial reduction in the required data rate up to 70.8% and a reduction in ATE by 53.7% using three agents and collaborative mapping compared

to standalone mapping. Hence, our system closes a gap by addressing the need for data-efficiency in collaborative visual SLAM setups. The feature compression can also be used when exchanging keyframes from a local visual odometry with the central server. Integrating such a lightweight odometry into the client for a local control loop is left for future work. Further evaluation and the source code are available online[1].

REFERENCES

[1] D. Van Opdenbosch, M. Oelsch, A. Garcea, and E. Steinbach, "Selection and Compression of Local Binary Features for Remote Visual SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7270–7277.

[2] J. Engel, V. Koltun, and D. Cremers, "Direct Sparse Odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[3] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2320–2327.

[4] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Direct Monocular SLAM," in *European Conference on Computer Vision (ECCV)*, vol. 8690, 2014, pp. 834–849.

[5] X. Gao, R. Wang, N. Demmel, and D. Cremers, "LDSO: Direct Sparse Odometry with Loop Closure," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2198–2204.

[6] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 15–22.

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

[8] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007, pp. 225–234.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[10] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

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

[12] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular SLAM with multiple Micro Aerial Vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3963–3970.

[13] L. Riazuelo, J. Civera, and J. M. Montiel, "C2TAM: A Cloud framework for cooperative tracking and mapping," *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 401–413, 2014.

[14] P. Schmuck and M. Chli, "Multi-UAV collaborative monocular SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3863–3870.

[15] M. Karrer, P. Schmuck, and M. Chli, "CVI-SLAM-Collaborative Visual-Inertial SLAM," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2762–2769, 2018.

[16] T. Cieslewski, S. Lynen, M. Dymczyk, S. Magnenat, and R. Siegwart, "Map API - Scalable decentralized map building for robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 6241–6247.

[17] M. Gadd and P. Newman, "Checkout my map: Version control for fleetwide visual localisation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 5729–5736.

[18] H. S. Park, Y. Wang, E. Nurvitadhi, J. C. Hoe, Y. Sheikh, and M. Chen, "3D Point Cloud Reduction Using Mixed-Integer Quadratic Programming," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2013, pp. 229–236.

[19] M. Dymczyk, S. Lynen, T. Cieslewski, M. Bosse, R. Siegwart, and P. Furgale, "The gist of maps - Summarizing experience for lifelong localization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2767–2773.

[20] M. Dymczyk, S. Lynen, M. Bosse, and R. Siegwart, "Keep it brief: Scalable creation of compressed localization maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 2536–2542.

[21] D. Van Opdenbosch, T. Aykut, M. Oelsch, N. Alt, and E. Steinbach, "Efficient Map Compression for Collaborative Visual SLAM," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, Lake Tahoe, USA, 2018, pp. 992–1000.

[22] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-Efficient Decentralized Visual SLAM," in *IEEE International Conference of Robotics and Automation (ICRA)*, 2018, pp. 2466–2473.

[23] T. Cieslewski and D. Scaramuzza, "Efficient Decentralized Visual Place Recognition Using a Distributed Inverted Index," *IEEE Robotics and Automation Letters (RAL)*, vol. 2, no. 2, pp. 1–8, 2017.

[24] ——, "Efficient Decentralized Visual Place Recognition From Full-Image Descriptors," in *IEEE International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2017, pp. 78–82.

[25] D. Tardioli, E. Montijano, and A. R. Mosteo, "Visual data association in narrow-bandwidth networks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 2572–2577.

[26] L. Baroffio, M. Cesana, A. Redondi, M. Tagliasacchi, and S. Tubaro, "Coding visual features extracted from video sequences," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2262–2276, 2014.

[27] L. Baroffio, A. Canclini, M. Cesana, A. Redondi, M. Tagliasacchi, and S. Tubaro, "Coding Local and Global Binary Visual Features Extracted from Video Sequences," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3546–3560, 2015.

[28] L. Bondi, L. Baroffio, M. Cesana, A. Redondi, and M. Tagliasacchi, "Multi-view coding of local features in visual sensor networks," in *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2015, pp. 1–6.

[29] A. E. Redondi, L. Baroffio, M. Cesana, and M. Tagliasacchi, "Multi-view coding and routing of local features in Visual Sensor Networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2016, pp. 1–9.

[30] D. Van Opdenbosch, M. Oelsch, A. Garcea, and E. Steinbach, "A Joint Compression Scheme for Local Binary Feature Descriptors and their Corresponding Bag-of-Words Representation," in *IEEE Conference on Visual Communications and Image Processing (VCIP)*, 2017, pp. 1–4.

[31] L.-Y. Duan, V. Chandrasekhar, J. Chen, J. Lin, Z. Wang, T. Huang, B. Girod, and W. Gao, "Overview of the MPEG-CDVS Standard," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 179–194, 2016.

[32] P. P. De Gusmao, S. Rosa, E. Magli, S. Lepsoy, and G. Francini, "Loop detection in robotic navigation using MPEG CDVS," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2015, pp. 1–6.

[33] L.-Y. Duan, V. Chandrasekhar, S. Wang, Y. Lou, J. Lin, Y. Bai, T. Huang, A. C. Kot, and W. Gao, "Compact Descriptors for Video Analysis: the Emerging MPEG Standard," *arXiv:1704.08141*, 2017.

[34] S. Lynen, T. Sattler, M. Bosse, J. Hesch, M. Pollefeys, and R. Siegwart, "Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization," in *Robotics: Science and Systems XI*, 2015.

[35] L. Baroffio, A. Redondi, M. Tagliasacchi, and S. Tubaro, "A survey on compact features for visual content analysis," *APSIPA Transactions on Signal and Information Processing*, vol. 5, 2016.

[36] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2564–2571.

[37] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, p. 629, 1987.

[38] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Y. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1–7, 2016.

[39] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361.

[40] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 573–580.

[41] L. Baroffio, A. Canclini, M. Cesana, A. Redondi, and M. Tagliasacchi, "Briskola: BRISK optimized for low-power ARM architectures," in *IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 5691–5695.

[1] https://rebrand.ly/ral18