ETH zürich

ETH Library

Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments

Conference Paper

Author(s):
Bartolomei, Luca; Karrer, Marco; Chli, Margarita

Publication date:
2020

Permanent link:
https://doi.org/10.3929/ethz-b-000441280

Rights / license:
In Copyright - Non-Commercial Use Permitted

Originally published in:
https://doi.org/10.1109/IROS45743.2020.9341367

This page was generated automatically upon download from the ETH Zurich Research Collection.
For more information, please consult the Terms of use.

Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments

Luca Bartolomei, Marco Karrer and Margarita Chli
Vision For Robotics Lab, ETH Zürich, Switzerland

***Abstract*— In this work, we present a system architecture to enable autonomous navigation of multiple agents across user-selected global interest points in a partially unknown environment. The system is composed of a server and a team of agents, here small aircrafts. Leveraging this architecture, computationally demanding tasks, such as global dense mapping and global path planning can be outsourced to a potentially powerful central server, limiting the onboard computation for each agent to local pose estimation using Visual-Inertial Odometry (VIO) and local path planning for obstacle avoidance. By assigning priorities to the agents, we propose a hierarchical multi-robot global planning pipeline, which avoids collisions amongst the agents and computes their paths towards the respective goals. The resulting global paths are communicated to the agents and serve as reference input to the local planner running onboard each agent. In contrast to previous works, here we relax the common assumption of a previously mapped environment and perfect knowledge about the state, and we show the effectiveness of the proposed approach in photo-realistic simulations with up to four agents operating in an industrial environment.**

I. INTRODUCTION

The growing popularity of the use of Unmanned Aerial Vehicles (UAVs) in tasks, such as exploration of unsafe areas, inspection, and search-and-rescue missions has been driving research towards their autonomous navigation. As aerial perception and path planning have become increasingly robust, small UAVs have been demonstrated to successfully plan their path and fly autonomously in some scenarios (e.g. [1]). However, since computational power and payload become the limiting factors when planning a mission with small UAVs, crucial choices regarding the sensor suite to be carried, as well as the type of algorithms that can be run onboard, need to be made in order to remain within the constraints of the platforms, such as battery lifetime. As a result, multi-robot collaboration is often considered, aiming to coordinate multiple agents to complete the mission within a limited time or resources budget, for example in search-and-rescue, investigation of spatio-temporal phenomena and inspection of hazardous areas [2].

For multi-robot collaboration to be effective, co-localization of these robots in a common map and coordination of their motion to avoid collisions are imperative. While multi-robot path planning has been receiving considerable attention in the literature with dedicated studies dating back at least three decades, there still is a lack of practical approaches usable in real scenarios as most existing methods rely on overly optimistic assumptions. For example, in reactive path planning [3] the pose of the robot is often assumed to be not subject to uncertainty, while map-based methods such as [4], [5] rely on known global maps and attempt to generate optimal paths through local maps. However, in real missions the map of the environment is most often unknown *a priori* and all state estimates are subject to drift. In [4], the authors tackle this issue by building a globally consistent map of the environment of interest using bundle adjustment. Nonetheless, their approach is limited to a single robot and requires an initial, manually piloted flight to construct the map used for planning and re-localization.

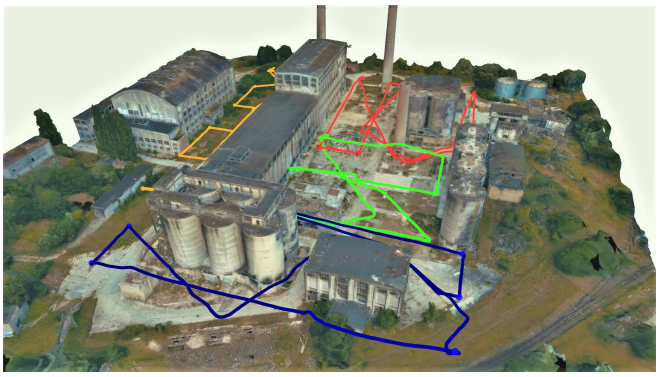


Fig. 1: 3D-view of the proposed system guiding safe and successful navigation of a team of four UAV-agents (with trajectories shown in different colors) in a photo-realistic simulation of an industrial site, despite that the user-defined waypoints for the UAVs correspond to partially overlapping regions. Via hierarchical planning, the proposed system is able to safely plan the UAVs' paths, running state estimation and local obstacle avoidance onboard each UAV, and sensor fusion and multi-robot global planning on the server to collect all UAVs' experiences in a joint, optimized map.

The impact of having unknown or only partially known maps and state estimation drift grows noticeably in multi-robot applications, where the agents should not only steer away from obstacles, but also need to avoid collisions between them. The coordination of multiple robots and the assignment of the flight area in a multi-UAV mission is not trivial, as with an increasing number of agents the state grows quickly. A typical approach from the literature is based on coordination diagrams, which are searched for paths by minimizing a global performance cost function [6], [7]. In [8] different priorities are set for the robots, while more complex approaches [9] explicitly consider the velocity-time space for coordinating robot motions. Other approaches use

This work was supported by Swiss National Science Foundation (SNSF, Agreement no. PP00P2183720), NCCR Robotics, Armasuisse and the HILTI group.

The open-source code is available at `https://v4rl.ethz.ch/research/datasets-code.html`. The video showing the experiments is available at `https://youtu.be/BlFbiuV-d10`

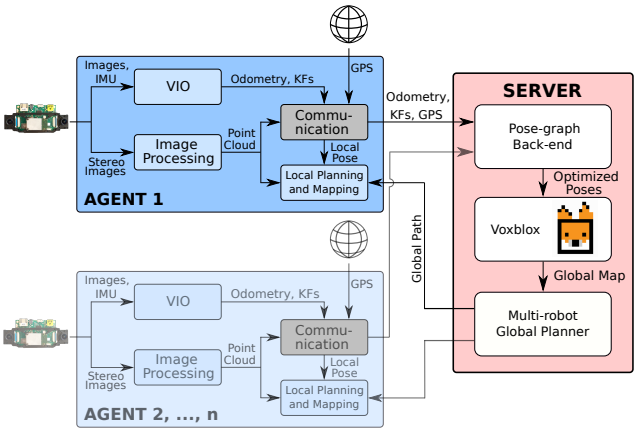


Fig. 2: The proposed architecture, composed of a set of agents $\{1, \cdots, n\}$ and a server. The agents process the sensor inputs (images, IMU and GPS measurements), perform state estimation in the VIO module and send the information alongside the odometry and the keyframes (KFs) to the central server. The server fuses these measurements in a pose-graph from which globally consistent poses in a common reference can be obtained. These poses, together with the compressed stereo point clouds, are fed to the Voxblox framework [13] to build a dense representation of the explored area, enabling the multi-robot global planner to compute paths which are sent back to the agents for execution.

mixed integer programming models to encode the interactions between robots [10], while others make use of grid search, roadmaps [11] and sampling-based planning [12]. Recent works [2] show promising results in multi-robot coverage planning, but they do not deal with the localization of multiple agents in a common map.

In this work, we take inspiration from existing approaches aforementioned, combining them in a new architecture composed of a central server and a team of robots (i.e. agents). The server collects the experiences of all robots and performs optimization-based sensor fusion to obtain their poses in a common GPS-based reference frame building up a joint, global dense map of the environment. This map is then used to generate global paths in a hierarchical fashion in order to coordinate the agents.

The objective of the proposed pipeline is to provide a practical solution for applications in search-and-rescue scenarios. Here, we assume that prior knowledge about the area of interest is available (e.g. from satellite images) and that a human operator assigns a list of goal positions to each robot in the team.

In brief, the contributions of this work are the following:

- the design of a centralized agent-server architecture, enabling merging of data from multiple agents (here UAVs), reducing the computational load of each agent,
- the design of a hierarchical planning strategy for multi-agent coordination in partially unknown space,
- an extensive evaluation of the proposed system in photo-realistic simulations with up to four UAV-agents, and
- the source code of the proposed system.

II. METHODOLOGY

The proposed system architecture is composed of a central server and a team of robotic agents, as illustrated in Fig. 2, with the aim of coordinating this team to explore an area of interest. Each agent is assumed to be capable of estimating its egomotion running keyframe-based Visual-Inertial Odometry (VIO) onboard – in our experiments, an adaptation of VINS-Mono [14] is used. As agents here are small rotorcraft UAVs, in order to follow a global path and avoid smaller obstacles, we employ the local planning approach proposed by [5], while the required local occupancy map is built up using a front-looking stereo camera onboard each agent. To limit the bandwidth requirement of communicating dense scene information to the server, these stereo point clouds are compressed by fusing the ones captured from subsequent frames via voxel filtering. The resulting point clouds are then referenced with respect to an anchor keyframe and are sent together with the keyframe information (including pose, keypoints and the 3D landmarks visible in the keyframe) and the GPS readings corresponding to that keyframe to the server.

The server performs some of the computationally more expensive tasks, such as optimization-based sensor fusion, mapping and global path planning. In particular, the VIO keyframe poses get fused with the GPS information in order to estimate each UAV's trajectory in a global reference frame, estimating the drift of the VIO of each agent. Using the result from the sensor fusion, the compressed point clouds are used to build up a global dense map of the environment using Voxblox [13]. Based on the resulting dense map, a hierarchical global path-planning strategy is employed to coordinate the mission, such that each agent reaches safely its goals in the previously unmapped area. In the following, the individual parts of the system are described.

A. Notation

In this paper, capital letters denote coordinate frames (e.g. A), bold capitals ($\mathbf{A}$) denote matrices and bold small letters ($\mathbf{a}$) vectors. Rigid body transformations from coordinate frame B to A are denoted as $\mathbf{T}_{AB}$, while the translational part of any transformation $\mathbf{T}$ is denoted by $\mathbf{p}$ and the rotational part as $\mathbf{R}$. For notational brevity, we use $\mathbf{T}_{AB} \cdot \mathbf{v}$ to denote the transformation of the vector $\mathbf{v}$ from B to A.

B. Pose-graph Back-end

In order to establish the relationships between the different types of measurements as well as amongst multiple agents, we use four different types of coordinate frames as illustrated in Fig. 3. The world frame W represents the common GPS reference frame and is unique across all agents. For every agent i we have a map frame M_i, representing the origin of that agent's drift-corrected map, which relates to W by a fixed transformation $\mathbf{T}_{WM_i}$. Finally, the frame O_i denotes the origin of the local VIO estimates describing the pose of the IMU (frame S_i). The drift of VIO is expressed as a time-varying transformation $\mathbf{T}_{M_iO_i}(t)$, where t denotes the time. Since in VIO systems the roll and pitch angles are

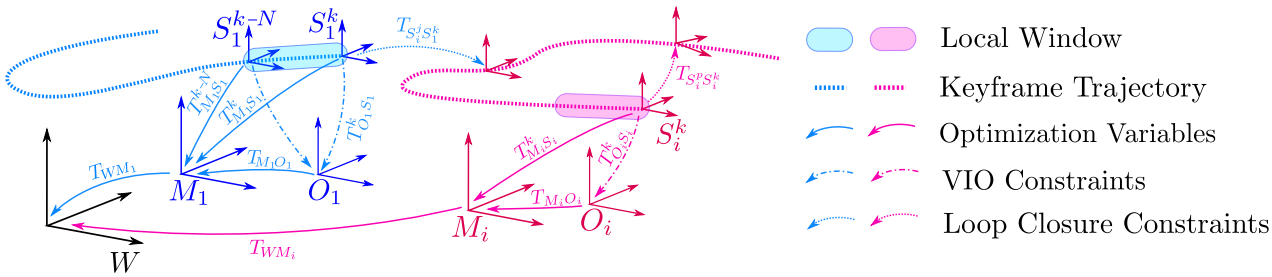


Fig. 3: Schematic depicting the transformation chains used in the proposed system. The world frame W denotes the GPS-reference frame and is shared amongst all agents, while every agent i has a map with origin M_i and an origin O_i for the corresponding VIO estimate. The body of agent i is indicated with the IMU frame S_i. The drift of the VIO of each agent is encoded in the transformation $\mathbf{T}_{M_iO_i}$. Every time a new keyframe gets inserted in the estimation of an agent, local optimization is triggered operating over the last N keyframes (indicated by the shaded region).

observable, as proposed in [14], only the position and the yaw angles of each keyframe are optimized. For the sake of readability, in the following we drop the agent index i when referring to transformations within a single agent.

1) Parameterization and Residuals: In order to optimize the desired transformations, as indicated in Fig. 3, we need to be able to express the measurements in terms of these transformations to form residuals. The first type of residual used is a prior, which we define as

$$\mathbf{r}_{AB} = \mathbf{T}_{AB} \boxminus \tilde{\mathbf{T}}_{AB}, \tag{1}$$

where $\tilde{\mathbf{T}}_{AB}$ represents the prior knowledge of the transformation $\mathbf{T}_{AB}$. The notation $\boxminus$ indicates a generalized subtractions, which in case of the used 4-DoF parameterization corresponds to

$$\mathbf{T}_1 \boxminus \mathbf{T}_2 := \begin{bmatrix} \mathbf{R}_2^T(\mathbf{p}_1 - \mathbf{p}_2) & \phi(\text{yaw}(\mathbf{T}_1) - \text{yaw}(\mathbf{T}_2)) \end{bmatrix}^T, \tag{2}$$

where $\text{yaw}(\mathbf{T})$ represents the yaw angle encoded in the transformation $\mathbf{T}$ and the function ϕ unwraps the yaw angles to lie within the range $[-\pi, \pi)$. The second type of residuals used are relative odometry constraints, representing the error between the measured relative pose of the keyframes j and p (estimated by the VIO) and the predicted transformation based on the state variables:

$$\mathbf{r}_{rel}^{j,p} = (\mathbf{T}_{MS}^j)^{-1}\mathbf{T}_{MS}^p \boxminus (\mathbf{T}_{OS}^j)^{-1}\mathbf{T}_{OS}^p \ . \tag{3}$$

Furthermore, we define a residual to relate the current state of $\mathbf{T}_{MO}$ together with the keyframe poses $\mathbf{T}_{MS}^j$ to the VIO poses ($\mathbf{T}_{OS}^j$) as follows:

$$\mathbf{r}_o^j = (\mathbf{T}_{MO})^{-1}\mathbf{T}_{MS}^j \boxminus \mathbf{T}_{OS}^j \ . \tag{4}$$

2) Initial GPS Alignment: In order to bootstrap the estimation of the transformation $\mathbf{T}_{WM}$ between the VIO map of one of the agents and the GPS reference frame, we align the VIO poses to the GPS measurements using least squares. The obtained transformation then gets refined in a non-linear optimization using Gauss-Newton. To decide whether the initialization was successful or not, we compute the covariance of the obtained transformation. As the translational part of $\mathbf{T}_{WS}$ can be estimated even without motion, we consider the system initialized only if

$$\sigma_{\text{yaw}} < \text{threshold} \tag{5}$$

holds, while σ_{yaw} is the marginal uncertainty of the estimated yaw angle.

3) Local Optimization: In the local optimization for agent i the poses of the most recent N keyframes (i.e. $\mathbf{T}_{M_iS_i}$), as well as the transformation between the Map and the GPS-reference frame $\mathbf{T}_{WM_i}$ are refined. Furthermore, the current drift of the VIO (i.e. $\mathbf{T}_{M_iO_i}$) is estimated. The local optimization runs independently for every agent and in our implementation is executed in separate threads. The objective of the local optimization is given by

$$\mathcal{X}_i^k = \text{argmin} \sum_{j=k-N}^{k} \Big(\|\mathbf{r}_{o_i}^j\|_{\Sigma_{o_i}}^2 + \sum_{p \in \mathcal{N}_j, p<j} \|\mathbf{r}_{rel}^{j,p}\|_{\Sigma_{rel}}^2 + e_{GPS}^j \Big) + \|\mathbf{r}_{WM_i}^k\|_{\Sigma_{WM_i}}^2 + \|\mathbf{r}_{M_iO_i}\|_{\Sigma_{M_iO_i}}^2, \tag{6}$$

where $\mathcal{X}_i^k$ represents the set of involved transformations:

$$\mathcal{X}_i^k := \begin{bmatrix} \mathbf{T}_{M_iS_i}^{k-N} & \cdots & \mathbf{T}_{M_iS_i}^k & \mathbf{T}_{WM_i} & \mathbf{T}_{M_iO_i} \end{bmatrix}. \tag{7}$$

The notation $\|\cdot\|_\Sigma^2$ denotes the squared Mahalanobis distance with covariance Σ. The set $\mathcal{N}_j$ denotes all connected neighbours of the keyframe j, which in our case corresponds to a fixed number of temporal neighbours. The terms $\mathbf{r}_{WM_i}^k$ and $\mathbf{r}_{M_iO_i}$ are the prior residuals associated to $\mathbf{T}_{WM_i}$ and $\mathbf{T}_{M_iO_i}$, respectively, while the error term $e_{GPS_i}^j$ summarizes all n_j GPS measurements associated to keyframe j based on the temporal proximity. In order to have a finer temporal resolution for the data association, we utilize the VIO pose of frame f relative to keyframe j to associate the GPS measurements to the keyframe poses:

$$e_{GPS_i}^j = \sum_{g=1}^{n_j} \|\mathbf{T}_{WM_i}\mathbf{T}_{M_iS_i}^j(\mathbf{T}_{O_iS_i}^j)^{-1}\mathbf{T}_{O_iS_i}^f\mathbf{p}_{S_iU} - \mathbf{p}_{GPS_i}^g\|_{\Sigma_{GPS_i}}^2, \tag{8}$$

where $\mathbf{p}_{S_iU}$ is the offset of the GPS antenna expressed in the IMU frame S_i and $\mathbf{p}_{GPS_i}^g$ corresponds to the GPS measurement g. After the optimization we recover the marginal covariances Σ_{WM_i} and $\Sigma_{M_iO_i}$ corresponding to $\mathbf{T}_{WM_i}$ and $\mathbf{T}_{M_iO_i}$, respectively, and use these as priors in the next optimization step.

4) Loop-Closure Detection: In order to establish additional constraints within and across the agents' trajectories, we perform visual loop-closure detection in a similar fashion as in [14] using the bag of binary words DBoW2 [15]. Since we want to enable loop detection also across the agents, a single database of words shared amongst all agents is maintained. New loop-closures are detected by first querying the database for visually similar candidates and the best Q candidates are subjected to descriptor-based 2D-2D brute force correspondence search. Any matches get checked for their associated 3D landmarks, which are reprojected from the query frame into the candidate frame and vice-versa, followed by a 3D-2D RANSAC outlier rejection. If sufficient inliers are found, the relative pose, e.g. indicated by $\mathbf{T}_{S_i^j S_1^k}$ and $\mathbf{T}_{S_i^p S_i^k}$ in Fig. 3, in case of loop detection between different agents or within the same trajectory, respectively, is optimized by minimizing the reprojection error of the established correspondences.

5) Global Optimization: Upon detection of a loop-closure, both within one agent's trajectory and across multiple agents' trajectories, we perform a global optimization refining all poses and map transformations that are connected. For example, in a 3-agent team (with labels $1, 2, 3$) let agents 1 and 3 have loop-closures between them. If an additional loops get detected within either 1 or 3's trajectory, all transformations associated with 1 and 3 get optimized, while the transformations within 2 remains independent. The objective of this global optimization step, is given by

$$\begin{aligned}\mathcal{X} = \operatorname{argmin} \sum_{i \in \mathcal{M}} \Big(\sum_{j \in \mathrm{M}_i} e^j_{GPS_i} + \sum_{p \in N_j, p<j} \|\mathbf{r}^{j,p}_{rel_i}\|^2_{\Sigma_{rel_i}} \\ + \sum_{p=k-N}^{k} \|\mathbf{r}^p_{o_i}\|^2_{\Sigma_{o_i}} \Big) + \sum_{k_i, p_j \in \mathcal{L}} \|\mathbf{r}^{k_i,p_j}_{lc}\|^2_{\Sigma_{lc}},\end{aligned} \tag{9}$$

where $\mathcal{M}$ denotes the set of all previously connected maps, $\mathcal{L}$ denotes the set of loop closure constraints between keyframe k of agent i and keyframe p of agent j. The optimization variables $\mathcal{X}$ are given by

$$\mathcal{X} := \begin{bmatrix} \mathcal{X}_1 & \mathcal{X}_2 & \cdots & \mathcal{X}_{|\mathcal{M}|} \end{bmatrix}, \tag{10}$$

where

$$\mathcal{X}_i := \begin{bmatrix} \mathbf{T}_{WM_i} & \mathbf{T}_{M_i O_i} & \mathbf{T}_{M_i S_i^1} & \cdots & \mathbf{T}_{M_i S_i^k} \end{bmatrix} \ \forall i \in \mathcal{M}, \tag{11}$$

while the loop closure residual $\mathbf{r}^{k_i,p_j}_{lc}$ is calculated as

$$\mathbf{r}^{k_i,p_j}_{lc} = (\mathbf{T}_{WM_i}\mathbf{T}_{M_i S_i^k})^{-1}\mathbf{T}_{WM_j}\mathbf{T}_{M_j S_j^p} \boxminus \tilde{\mathbf{T}}_{S_i^k S_j^p}, \tag{12}$$

where $\tilde{\mathbf{T}}_{S_i^k S_j^p}$ is the relative pose obtained as outlined in Sec. II-B.4.

C. Mapping and Multi-robot Global Path Planning

The results of the optimization of the pose-graph in the back-end are used to create a global, dense map of the area in the world reference frame by means of the Voxblox framework [13]. In order to construct the map, we use the compressed point clouds as communicated by the agents together with the optimized pose of the corresponding anchor keyframe.

The updated information about the space is utilized by the multi-robot global path planner in order to coordinate the movements of the UAVs. At the beginning of a mission, every agent is assigned an area to cover as an ordered sequence of GPS waypoints, which have to be reached as quickly as possible. The planner computes the paths for all the agents in the global map and communicates them to the respective agents. Given the current map, any unreachable waypoints (e.g. placed inside an obstacle) are discarded by the planner. Upon completion of the exploration mission, each UAV returns to its respective home position (e.g. where the planning started) by planning only in explored and known space.

To compute the paths, we employ the standard version of RRT* [16] implemented in the OMPL library [17]. The global planner optimizes the path lengths and is optimistic, i.e. it considers unknown space to be free and uses the Truncated Signed Distance Field (TSDF) information available in the Voxblox map. Nonetheless, when a robot has to return to the starting position, the global planner adopts a pessimistic behavior, i.e. planning happens only in known free space. To avoid collisions amongst the robots, we propose a hierarchical approach for multi-robot coordination. At startup, we arbitrarily assign a priority level to each agent, considering that in the team there cannot be two agents with the same priority. The planning happens hierarchically, meaning that the possible flight area of an agent is constrained by both the global occupancy map and the global paths of the agents with higher priority levels. In other words, given the total space V_{TOT}, we remove the occupied parts V_o to obtain the available obstacle-free volume $V_f = V_{TOT} \setminus V_o$. Given n agents, the available space for an agent i is the free space minus the area of flight of all agents with higher priority than i's, i.e.

$$V_i = V_f \setminus \bigcup_{j \in higher\ priority} V_j, \qquad i \in \{0, \cdots, n\}\ . \tag{13}$$

During the planning, we consider a safety bound of dimension d around the paths, while the agents are modeled as spheres with radius r, with $d > r$. The planner starts by computing the trajectory for the agent with highest priority, whose flight space is subjected solely to obstacles. The planner then sequentially proceeds to plan for lower priority levels, until all agents have estimated a valid global path. In order to be reactive to changes in the global map, the global planner checks all the paths for collisions with the updated map at a fixed rate. We consider a path to be invalid if it is in collision with an obstacle or with another robot with higher priority. In case a path for one of the UAVs is invalid, the global planner re-plans the corresponding trajectory and performs collision checking on the paths of the other lower priority UAVs; all the invalid paths will be re-computed.

Fig. 4: Top view of the simulation with four agents. Four different regions were selected and each arbitrarily assigned to one of the agents. As the waypoints were chosen to form a lawnmower pattern, some of them lie inside obstacles (e.g. buildings). The global planner successfully identified and skipped these positions while reaching the accessible remaining points of interest.

D. Local Path Planning and Obstacle Avoidance

On each agent, we run a local planner adapted from the planning strategy of [5]. The trajectories are represented as quintic Uniform B-Splines allowing to ensure smoothness up to the snap, while to perform obstacle avoidance a local occupancy model of the environment centered at robot position is maintained. The shape of the B-Spline of order k is locally determined by a set of $k+1$ control points $\mathbf{p}_i,\ i \in [0, \cdots k]$. Using a set of equitemporal control-points, the authors of [5] pose the problem of local trajectory re-planning as an optimization problem with the following cost function:

$$E_{total} = E_{ep} + E_c + E_q + E_l \ , \tag{14}$$

where E_{ep} is an endpoint cost function penalizing position and velocity deviations at the end of the optimized trajectory segment from the desired ones from the global path; E_c is a collision cost function, E_q the cost of the integral over the squared derivatives (acceleration, jerk, snap) and E_l is a soft limit on the norm of the derivatives over the trajectory. In order to initialize the control points, we utilize the trajectory samples of the global path. However, as the global path is expressed in W, while the reference frame for the controller of agent i is O_i, the global trajectory is transformed using the most recent received estimate of $\mathbf{T}_{WO_i}$ as computed in the sensor fusion back-end. In order to allow for changes in the global paths when a global re-planning action is triggered on the server, we adopt a receding-horizon approach, by planning at a fixed distance along the global path from the current agent's position.

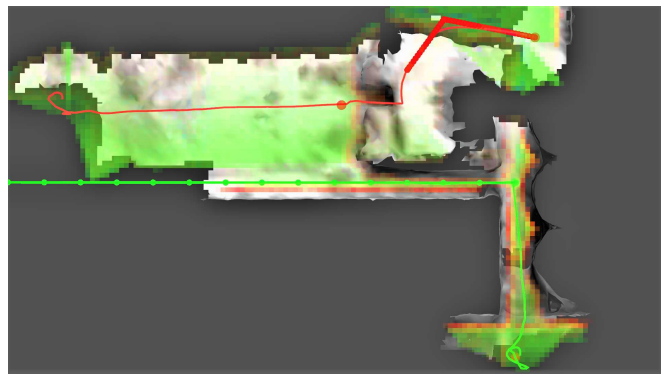
III. EXPERIMENTAL RESULTS

A. Benchmarking of Pose-graph Back-end

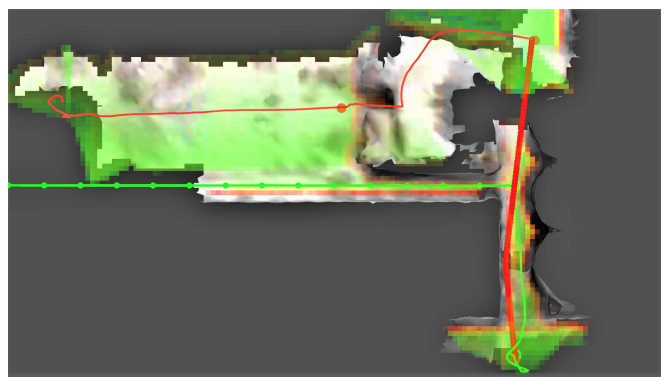
To the best of our knowledge, there is no available dataset containing high-quality visual-inertial data, GPS data and ground truth, so we evaluate the proposed system on the Machine Hall (MH) sequences of EuRoC dataset [18], simulating GPS measurements by disturbing the available global position measurements with Gaussian noise of a standard deviation of $0.15m$. As these position measurements are expressed in the ground-truth frame, this allows us to evaluate the quality of the estimated map-to-world transformation as well.

The resulting Absolute Trajectory Errors (ATEs) of the keyframe poses are shown in Table I. From the reported values, it can be seen that the proposed sensor fusion approach is successfully able to eliminate the drift in the estimation. In

(a) Top-view of the experiment with the paths followed by the agents.



(b) Map before global planning step for Agent 2 (in red).



(c) Planning result for Agent 2.

Fig. 5: Experiment with two agents assigned to overlapping areas of interest. In (a) a top-view of the experiment is shown, where the paths executed by Agents 1 and 2 are shown in green and red, respectively. (b) shows the situation before the re-planning step of Agent 2. The colored field indicates the traversable space computed by Voxblox, where the color changes from green to red in the direction towards obstacles. (c) shows the re-use of the map created by Agent 1 enabling Agent 2 to plan a shorter path through the same alley.

Dataset	VIO only RMSE $[m]$	Optimized RMSE $[m]$	Optimized* RMSE $[m]$
MH01_easy	0.146	0.061	0.073
MH02_easy	0.238	0.082	0.095
MH03_medium	0.210	0.098	0.104
MH04_difficult	0.330	0.092	0.097
MH05_difficult	0.305	0.115	0.118
MH01-MH05	-	0.099	0.101

TABLE I: ATEs of the keyframe poses averaged over three runs as they enter the proposed system using VIO only, the resulting output of the proposed back-end optimization, and marked with a '*' the ATEs using the estimated $\mathbf{T}_{WM}$.

Message Type	Mean Bandwidth	Std Deviation
Keyframes	24.91 KB/s	16.05 KB/s
Point clouds	68.78 KB/s	35.35 KB/s

TABLE II: Bandwidth consumption statistics for the communication between one agent and the central server in the experiment described in III-B.1. The network traffic necessary to send GPS information to the server and to communicate the paths computed by the global planner to the agents is negligible.

order to evaluate the quality of the estimated transformations between the map(s) and the world frame, the last column in Table I reports the ATE of the estimated trajectories aligned to the ground-truth using the final estimate of the corresponding transformations $\mathbf{T}_{WM_i}$. As it can be seen, the ATE is only marginally increased when compared to the ATE using least squares trajectory alignment, indicating a high accuracy of the estimated transformations.

B. Experiments in Photo-realistic Simulations

The proposed system has been extensively tested in photo-realistic simulations using the Gazebo and RotorS frameworks [19]. The environment used is a reconstruction of a chemical plant in Rüdersdof, Berlin[1]. A 3D-view of the model, spanning an area of $120m \times 120m$, is shown in Fig. 1. In order to demonstrate the main capabilities of the proposed system, three different experiments have been carried out:

1) Autonomous navigation along user-defined waypoints with four agents inside the 3D model,
2) Map re-use for two agents with overlapping areas of interest, and
3) Hierarchical planning for a team of three agents operating within the same area.

In the following, the experiments are described in detail. Note that all results can be visualized in the accompanying video at `https://youtu.be/BlFbiuV-d10`

1) Map navigation with multiple agents: In this experiment we show the intended use-case of the proposed system, guiding the safe and successful navigation of a team of four UAVs in a previously unknown area given an ordered list of user-defined points of interest (i.e. waypoints), as shown in Fig. 4. The waypoints here were selected by dividing up the space amongst the agents and sampling the corresponding area to form a lawnmower pattern. All agents successfully reached all accessible points of interest, while any ill-placed waypoints (e.g. inside or too close to buildings) get discarded by the global planner during navigation. After reaching the last point of interest, all agents navigate successfully back to their starting positions, by planning exclusively in known free space. In Table II we report the bandwidth usage statistics due to the communication between one agent and the central server over the experiment. Thanks to the compression and the filtering of the point clouds as described in Sec. II, the average bandwidth consumption is maintained low.

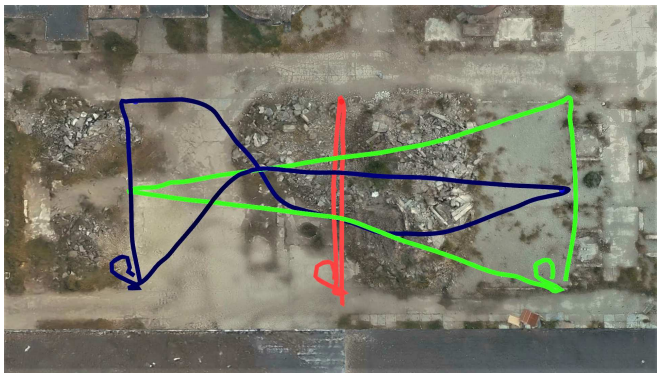
2) Map re-use between two agents with overlapping areas of interest: In this experiment, we showcase the advantage of the proposed centralized architecture. As the dense map contains all agents' information, parts mapped by one agent can be re-used for planning by another agent, as shown in this experiments with two UAVs operating in overlapping parts of the model. In Fig. 5b, the first agent, shown in green, navigates through a small alley in order to reach its first waypoint. The second agent, shown in red, exploits the information gathered by the first agent, enabling it to take the short path through the same alley instead of navigating around the block, as illustrated in Fig. 5c.

3) Hierarchical planning for three agents within the same area of interest: In the last experiment, we demonstrate the proposed hierarchical approach of the multi-robot global planner. As depicted in Fig. 6a, the goal positions for the three agents are placed within the same navigation area. The path followed by the agent with highest priority is shown in red, while the trajectories of the agents with intermediate and lowest priority are represented in green and blue, respectively. Since the agent with highest priority has to reach two waypoints placed in front of it, it plans straight-line trajectories, while the other two agents have to compute intertwined paths. While the UAV in green has to consider only the red UAV, the area of flight of the blue UAV is considerably reduced by the presence of the other two robots. Nonetheless, all agents are able to reach the assigned goals avoiding collisions amongst them and with the obstacles in the map. In Fig. 6b we show a 3D-view of the planning step when the agents are about to reach their second waypoints. After the completion of the mission, all the agents return to their respective homing positions.

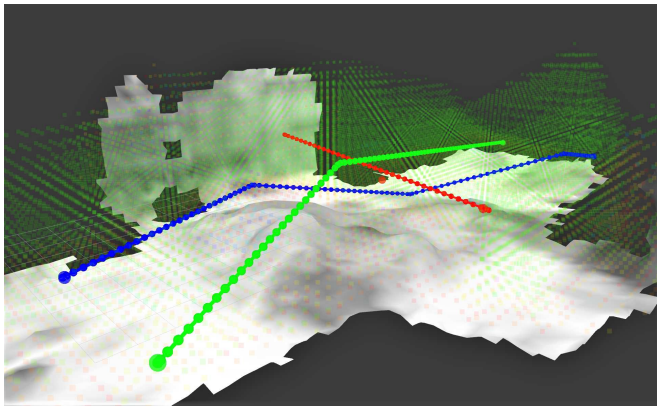
IV. CONCLUSIONS

In this paper, we propose a multi-robot estimation and coordination framework for autonomous navigation in partially unknown environments, given a sequence of user-defined points of interest. The system is demonstrated to guide the navigation of a team of UAVs as robotic agents, successfully reaching these waypoints, while ensuring no collisions amongst them or the scene. The UAVs share their experiences with each other via a central server, which creates a globally consistent map of the navigation environment using the proposed multi-agent back-end to bring all estimates in a common reference frame. This map is then used for multi-robot global planning running on the server, coordinating the movement of the team by communicating the computed paths to the agents. These then navigate along the paths, while

[1] `https://sketchfab.com/3d-models`



(a) Top-view of the experiment with the paths followed by the agents.



(b) 3D-view of the crossing paths, together with the traversable space and the mesh from Voxblox.

Fig. 6: Experiment with three agents flying in the same area. The sequence of waypoints is selected such that the paths of the agents cross each other as shown in (a). The UAV with highest priority, in red, has to reach two waypoints in front of it, while the other two UAVs, in green and in blue (lowest priority), need to navigate through the whole area in order to reach their goals. The UAVs successfully avoid collisions with each other and the scene as shown in (b).

performing state estimation and local obstacle avoidance onboard. This architectural design allows us to drop some of the typical assumptions commonly made in the multi-robot path-planning literature, such as perfect knowledge of the agents' poses and the map of the environment.

Furthermore, the source code of the system will be made publicly available.

Future directions will investigate the improvement of the system in terms of scalability and the use of automatic assignment of waypoints to agents following a pre-specified strategy (e.g. maximizing coverage) in a bid to boost the autonomy of multi-robot missions.

References

[1] L. Teixeira, I. Alzugaray, and M. Chli, "Autonomous Aerial Inspection using Visual-Inertial Robust Localization and Mapping," in *Field and Service Robotics*. Springer, 2018.

[2] T. Kusnur, S. Mukherjee, D. Mauria Saxena, T. Fukami, T. Koyama, O. Salzman, and M. Likhachev, "A planning framework for persistent, multi-UAV coverage with global deconfliction," *arXiv preprint arXiv:1908.09236*, 2019.

[3] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[4] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart, "Real-time visual-inertial mapping, re-localization and planning onboard MAVs in unknown environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[5] V. Usenko, L. von Stumberg, A. Pangercic, and D. Cremers, "Real-time trajectory replanning for MAVs using uniform B-Splines and a 3D circular buffer," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[6] L. E. Parker, "Path planning and motion coordination in multiple mobile robot teams," *Encyclopedia of complexity and system science*, 2009.

[7] P. Svestka and M. H. Overmars, "Coordinated path planning for multiple robots," *Robotics and Autonomous Systems (RAS)*, vol. 23, 1998.

[8] S. J. Buckley, "Fast motion planning for multiple moving robots," in *International Conference on Robotics and Automation (ICRA)*, 1989.

[9] J. van den Berg, J. Snape, S. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[10] E. J. Griffith and S. Akella, "Coordinating multiple droplets in planar array digital microfluidic systems," *The International Journal of Robotics Research (IJRR)*, vol. 24, 2005.

[11] M. Peasgood, C. M. Clark, and J. McPhee, "A Complete and Scalable Strategy for Coordinating Multiple Robots Within Roadmaps," *IEEE Transactions on Robotics (TRO)*, vol. 24, 2008.

[12] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," *The International Journal of Robotics Research (IJRR)*, vol. 35, 2016.

[13] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[14] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics (TRO)*, vol. 34, 2018.

[15] D. Gálvez-López and J. D. Tardós, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Transactions on Robotics (TRO)*, vol. 28, 2012.

[16] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research (IJRR)*, vol. 30, 2011.

[17] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine (RAM)*, vol. 19, 2012, http://ompl.kavrakilab.org.

[18] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research (IJRR)*, vol. 35, 2016.

[19] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "RotorS – A Modular Gazebo MAV Simulator Framework," in *Studies in Computational Intelligence*, vol. 625, 2016.