Visual trajectory learning and following in unknown routes for autonomous navigation

David A. Márquez-Gámez and Michel Devy

Abstract—This paper describes the design and testing of a system to enable large scale cooperative navigation of autonomous vehicles moving on a priori unknown routes. A large-scale learning-mapping approach and a map-based replay-localization method are combined to achieve cooperative navigation. The learning approach is based on a proposed hierarchical/hybrid BiCam SLAM and the replay approach exploits a localization method based on an active search procedure. The system can be generalized to be executed on multiple vehicles moving as a convoy. A global 3D map maintains the relationships between a series of local maps built by the first vehicle of the convoy (leader), defining a path that all other vehicles (followers) must stay on. Only single camera setups are considered. The overall approach is evaluated with real data acquired in an urban environment.

I. INTRODUCTION

Many robotic missions can be more efficiently and robustly achieved by a team of robots. This paper is focused on the cooperation of several vehicles, moving alone or as a convoy in open environments, on a priori unknown routes. Although most existing mobile robotic applications involve a single robot, a wide range of potential applications require multiple robots to execute joint tasks, e.g. rescue robotics, cooperative monitoring [1], [2], [3].

Localization is a key technology to address how the robots localize themselves in the operating unknown environment and how they know their local poses with respect to other objects. A variety of approaches have been reported for localization of multirobot formations. In [4], the localization problem of a leader-follower system, is based on EKF to estimate each follower’s pose with respect to the leader. In [5], it is proposed a behavior-based approach for maintaining formation of a team of robots, with experiments performed on outdoor unmanned ground vehicles equipped with vision, GPS and hazard sensors. In [6], a scheme for distributed outdoor localization for a team of robots is based on heterogeneous sensors such as local area differential global positioning system (LADGPS) and cameras. Most of the above approaches focused on the relative localization only, and a few of them discussed how to globally localize robots. Many methods simply assume that robots are equipped with absolute positioning capabilities.

We present a complete system for large scale autonomous operation of a team of robots in a “convoy” formation navigating in a dangerous, unknown and changing outdoor environment, typically on routes that could be mined. The proposed system consists of 2 steps: (1) learn first a safe path using an unmanned robot, and (2) follow this same path by other vehicles of the fleet. The two steps involve different functions: the initial path learning requires SLAM, i.e. to simultaneously memorize successive robot positions on the path and landmark positions that are used by the robot to locate itself. When the leader’s path is traveled again by other vehicles of the convoy, the map will be exploited to localize and control a vehicle so that it is maintained on the same path, with a tolerance that must be minimized.

It is assumed that all vehicles are only equipped by cameras and odometers, used during the learning or replay steps. The temporal gap between the two steps depends on the mission. (1) the learning and replay functions could be executed independently, i.e. the map and the path are acquired at time A, and the safe trajectory is followed again later at time B, with $B - A$ equal to several hours or days. In such a situation, the environment could change between the two steps: recorded landmarks could be removed during the interval, or the path itself could get blocked. Moreover vehicles used at time A and B could be the same, or could be different, involving the use of different cameras, so possibly different radiometric information. (2), these functions could be executed on different communicating vehicles navigating in a convoy formation. A leader vehicle records the map and the path, and sends these information to the second vehicle of the convoy using a wireless network; the first follower stays on the path taken by the leader, updating the map from its own observations and sends the updated map and the path to the next follower. Mutual localization (the follower sees the leader) is possible, but not mandatory. The leader and followers are equipped with different cameras, involving possible radiometric differences on images.

The contribution of this paper concern large scale navigation in a “convoy” formation in unstructured, three-dimensional terrain. Only perception is considered. Moreover, the environment is assumed to be static (no dynamic obstacle), but it can be modified between the learning and replay steps.

In section II the system description is presented. In section III the learning-mapping approach is described. In section IV the replay-localization method is presented. Section V gives experimental results for the proposed system using a real robot. Finally in section VI, current results and conclusions are discussed.

II. SYSTEM DESCRIPTION

The major processing blocks of our system are depicted in Fig. 1. The system consists of 2 steps: (1) the learning-mapping step, where a leader vehicle, builds several 3D-points maps and learn a safe path, and (2) the replay-localization step, where the information learned and trans-
mitted by the leader, allows other vehicles of the fleet, to follow the learned safe path. The two steps involve different functions: the initial path learning is based on a proposed hierarchical SLAM using only vision in a BiCam configuration and the replay procedure exploits a localization method based on an active search procedure to localize and control a vehicle so that it is maintained on the same path than during the learning step, with a tolerance that must be minimized. In sections III and IV learning and replay procedures are described.

![Fig. 1. An overview of the major processing blocks in our proposed system](image)

**III. LEARNING STEP: LEARN A SAFE TRAJECTORY**

**A. Hierarchical BiCam SLAM**

Our application involves motions of robots in large environments: so the mapping problem is formulated using submaps in a similar manner to hierarchical SLAM [7]. Independent consecutive local maps are represented in their submaps in a similar manner to hierarchical SLAM [7]. Environments: so the mapping problem is formulated using submaps in a similar manner to hierarchical SLAM [7].

![Fig. 2. The map: a set of submaps with their own lrf’s, and with common landmarks, known either by euclidian coordinates for close points (blue dots) or by IDP vectors generally for points at infinity (red X). Graphical model as a simplified Bayesian network: submaps constitute the local level (\(x_i, g_i\)) and the global level on top links the local submaps (W_i). The map building is sequential, corresponding to the robot exploring and not closing loops, having always \(w_{k+1} = x_k\).](image)

The principle is represented on Fig. 2.

1) **Local Level:** The local level contains the locally referred stochastic maps of landmarks, built with the BiCam SLAM algorithm [8]. Here a landmark is a 3D point, observed as an interest point in images. Each point is linked to an image patch which is defined as a small image region (traditionally 11x11 pixels). The \(k\)-th local map is defined by \(x_k^l = [x_k, g_k]^T\) (superscript “L” stands for local map) where \(x_k\) is the current pose of the robot, and \(g_k = [l_k^1 \cdots l_k^m]^T\) is the set of \(m\) mapped landmarks, both with respect to the \(k\)-th lrf. The camera and robot positions are linked by a known rigid transformation, so that BiCam SLAM can keep a Gaussian estimate \(x_k^L \sim N(\hat{x}_k, P_k^L)\) of this map, namely:

\[
\hat{x}_k = \begin{bmatrix} \hat{x}_k \\ \hat{g}_k \end{bmatrix}, \quad P_k^L = \begin{bmatrix} P_{x_k|x_k} & P_{x_k|g_k} \\ P_{g_k|x_k} & P_{g_k|g_k} \end{bmatrix}
\]

Maps are built sequentially. Once a threshold is reached, either in number of landmarks or in robot uncertainty, the current map \(x_k^L\) is closed and a new map \(x_{k+1}^L\) is created, starting in a new lrf with robot pose \(\hat{x}_{k+1}\) and error covariance equal to zero. Each landmark known in \(x_k^L\) is reobserved when the new map \(x_{k+1}^L\) is created, keeps the same label.

2) **Global Level:** The global level is represented as an adjacency graph in which local maps \(x_k^L\) in wrf are nodes, and the edges between them are annotated first by the relative transformations between successive lrf’s, noted \(w_k^{L+1}\), and secondly, by the list of labels of common landmarks, between \(x_k^L\) and \(x_{k+1}^L\). Let us define the global level as the Gaussian state \(\hat{w} \sim N(\hat{w}, P_w)\) of relative transformations between local maps, namely:

\[
\hat{w} = \begin{bmatrix} \hat{w}_1^1 \\ \vdots \\ \hat{w}_{k+1}^1 \end{bmatrix}, \quad P_w = \begin{bmatrix} P_{w_0} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & P_{w_{k+1}} \end{bmatrix}
\]

Let us note \(W_k\) the origin of the local map \(x_k^L\), expressed in the wrf; it can be computed by compounding relative transformations from \(w_0^1\) to \(w_{k+1}^1\). The global level can be viewed as a sparse pose-SLAM as in [9], where local maps are like landmarks hanging from robot poses in wrf. The global state in our case contains transformations \(w_{k+1}^1\), instead of absolute poses \(W_k\) (Fig. 2).

**B. Off line refinement of a map to be used later**

In the case of a path recorded in order to be run later, the final model is memorized as a hierarchical map. For an edge between nodes \(i\) and \(i+1\), a common landmark \(l\) has two representations \(l^{(i)}\) in the lrf \(i\), and \(l^{(i+1)}\) in the lrf \(i+1\). This model must be refined or transformed before to be exploited later for navigation.

1) **Refinement of the global map:** In our context, a path follows a route between two different areas, so that a classical Loop Closure is not possible; the global graph has no cycle. Nevertheless, the global map can be refined, by non linear optimization. Successive submaps can be made consistent, computing optimal and unique representations for all common landmarks and optimizing transforms between submaps.
We formulate the refinement of the global map as a local submap joining problem in a similar manner that in [10]. The algorithm is generalized and formulated as a least squares optimization problem and solved by Extended Information Filter (EIF) together with smoothing and iterations.

A local map is defined by equation (1). Since the local map provides a consistent estimate of the relative position from robot poses to local features, this map can be treated as an observation made from the robot start pose to all the features in the local map and a virtual robot located at the robot end pose. The observation value is the local map state estimate and the observation noise is a zero-mean Gaussian pdf with the covariance matrix equal to the local map covariance matrix. Thus, the refinement of the global map becomes a large-scale estimation problem with only local maps information.

The Extended Information Filter (EIF) is used to solve the estimation problem. A non zero off-diagonal element in the information matrix (a link between the two related objects) occurs only when the two objects are within the same local map. Since the size of each local map is limited, any object will only have links with its nearby objects no matter how many (overlapping) local maps are fused (Fig. 2). This results in an exactly sparse information matrix similar to Smoothing and Mapping (SAM) [11].

IV. Replay step: Replaying the Learnt Trajectory

Using the map database (a set of optimized local maps) produced by the leader robot, the followers robots are able to repeat a learned path any number of times. Two modes are considered: (1) in the replay mode the trajectory is globally stored in a database by the leader, and executed again long time after by another vehicle; (2) in the convoy mode, submaps are successively transmitted from the leader to the followers. Each follower re executes the path attached to each submap, and possibly refines the landmark positions, overall it can observe the previous vehicle of the convoy. During the execution, every 3D-point landmark of the map is sent to an active search procedure. If a landmark is not matched a number of times, then, it is removed from the map and replaced by another one initialized in the same image region.

A. Localization from the learned map

Before a vehicle replays a learnt path, it must be able to localize itself from an initial position close to that the leader initially had. Indeed, it is assumed that the follower vehicle starts from a position close to that of the leader but not necessarily identical. Then the follower must compare the map information with its own perception of the environment and compute its localization.

1) Initial position: In our approach, the follower vehicle has to localize itself and navigate thanks to a 3D map with accuracy and in real time. The initialization step is a critical point because the vehicle location is not accurately known in the map. In order to focus on its real localization, it perceives its environment through its camera. So, each image patch associated to the 3D point supposed to be observed is actively searched within an elliptical region. The latter is based on the projection of the uncertainty of the considered point taking into account the vehicle position. A cross-correlation measure based on ZNCC (Zero Normalized Cross Correlation) is used ([12]). While this search is performed, the best score for each image patch is stored, which provides a first hypothesis of matching. For each pair of 3D–2D points, an update step is calculated through a Kalman filter (details are explained in next section). Once the update, the difference between the estimated projection of each point of the map in the image and the result of the cross-correlation measure decreases. This allow the selection of a first 2D–3D couple.

2) On-line Localization: At this stage a good estimate of the starting point is available. Thanks to the fairly close estimate of the operating point, we can rely on a Kalman filter to estimate the position with proprioceptive sensors and refine it by using observations collected during the motion. This method is fully described and evaluated in [13].

2.1) Prediction step: The first step of the Kalman filter consists in the prediction through proprioceptive data. It establishes a model of evolution of the vehicle using a tricycle model. The equation model is:

\[
\begin{align*}
X_{k+1} &= X_k + ds\cos(\theta_k + \alpha/2) + \epsilon_x \\
Y_{k+1} &= Y_k + ds\sin(\theta_k + \alpha/2) + \epsilon_y \\
Z_{k+1} &= Z_k + ds\sin(\phi) + \epsilon_z \\
\theta_{k+1} &= \theta_k + \epsilon_\theta \\
\phi_{k+1} &= \phi_k + \epsilon_\phi \\
\psi_{k+1} &= \psi_k + \epsilon_\psi
\end{align*}
\]

where \(\theta, \phi, \psi\) are respectively the yaw, pitch and roll angles. The length \(ds\) is a function of a priori traveled distance and parameters which are specific to the vehicle. It is supplied by odometry measurements. The angle \(\alpha\) reflects the constant steering angle between times \(k\) and \(k+1\) and the parameters of the vehicle. The uncertainty in this estimate is provided by the Jacobians of the evolution model. The variance-covariance matrix is then expressed as, \(P_{k+1|k} = F_k P_k F_k^\top + Q_{k+1,k}\), where \(Q\) is the covariance of process noise and \(F_k\) is the Jacobian of the evolution model, with respect to \(X\).

2.2) Data association step: The search for correspondence is limited by the projection of the uncertainty associated with the movement of the vehicle and the uncertainty of the position of the 3D point. A RANSAC algorithm, which is similar in part to that described in [14], can reject absurd tracking results and find the correct matchings.

2.3) Estimation step: This prediction is then refined with the 2D observations of the 3D-world points. The association provides information on data matching between the points \((u_{obs}, v_{obs})^\top\) of the 2D image and 3D points \(P_3D\) of the map. Let us define the rigid transformation between the world and the reference frame of the vehicle as: \(R = R_c(\theta_k) R_y(\phi_k) R_z(\psi_k)\) and \(T = (X_k, Y_k, Z_k)^\top\). The 3D points of the map are projected in the image with the linear relationship, \(P_{2D} = KR_1(P_{3D} - T) = (ku, kv, k)^\top\), where \(K\) is the intrinsic parameter matrix of the camera.

The estimated coordinates are divided by the scale factor, \(u_{est} = ku/k\) and \(v_{est} = kv/k\).
\begin{equation}
\begin{aligned}
    u_{\text{ext}} &= h_u(P_{3D}) = \frac{K_u R (P_{3D} - T)}{K_u R (P_{3D} - T)} \\
    v_{\text{ext}} &= h_v(P_{3D}) = \frac{K_v R (P_{3D} - T)}{K_v R (P_{3D} - T)}
\end{aligned}
\end{equation}

where \(K_i\) is the \(i\)-th line from the intrinsic parameter matrix and \(h_u\), \(h_v\) are the part of the observation function related to the \(u\) and \(v\) coordinates.

The covariance of the innovation of the Kalman filter (\(H_k\)), is obtained using the Jacobians of the observation model. The Jacobians are calculated, using equation (4). The Kalman gain associated with a pair can be calculated by, \(G_{k+1} = P_{k+1|k} H_k^T (H_k P_{k+1|k} H_k^T + R_{\text{obs}})^{-1}\), where \(R_{\text{obs}}\) denotes the covariance of the noise associated with the observation, in pixels.

Finally, the data is updated from observation by:

\begin{equation}
X_{k+1|k+1} = X_{k+1|k} + G_{k+1} \left( \begin{pmatrix} u_{\text{obs}} \\ v_{\text{obs}} \end{pmatrix} - \begin{pmatrix} u_{\text{ext}} \\ v_{\text{ext}} \end{pmatrix} \right)
\end{equation}

\begin{equation}
P_{k+1|k+1} = P_{k+1|k} - G_{k+1} H_k P_{k+1|k}
\end{equation}

This step is for each pair of 3D-2D points. Localization accuracy depends on the number of points, the accuracy of the initial positioning and of the learnt map.

**B. Path tracker: trajectory following**

Path tracker module have been developed in a similar manner to Tiji [15], to integrate the replay step algorithm into our proposed system. Fig. 3, shows the path tracker processing block.

---

**Path tracking**

Path tracker is an algorithm that computes a feasible trajectory between a start and a goal state. The method proposed, which relies upon a parametric trajectory representation, is variational in nature. The trajectory parameters are incrementally updated in order to optimize a cost function involving the distance between the end of the trajectory computed and the desired goal. Should the goal state be unreachable (e.g. if the final time is ill-chosen), path tracker returns a trajectory that ends as close as possible to the desired goal.

1) **Trajectory planning:** This module provides the algorithm to plan and apply geometric transformation over a discrete trajectory. This module does not include any global motion planning approach as the current experiments have been focused on repeating a learnt trajectory.

2) **Trajectory following:** This module encapsulates the navigation algorithm. Taking as input the robot localization provided by the replay-localization and the discrete representation of the reference path provided by the learning step, this module provides a reactive method trying to follow a reference trajectory while providing at each time step a parametric continuous control trajectory (continuous sequence of input commands) until a given time horizon.

3) **Control trajectory:** Given the continuous control trajectory provided by the previous step, the Control trajectory module is aimed at checking if the control is still valid and admissible for the robotic system (respects its motion constraints) and computing a command to provide as input of the actuators.

---

**V. Experimental Results**

**A. Learning and Replay**

For this experiment, we have tested our system in the parking lot of the LAAS-CNRS (see Fig. 4) using a ground robot (Segway platform) equipped with two independent stereo-vision benches. The first one used in the Learning step is made of two Flea2 1280 x 960 cameras. The second one used in the Replay step is made of two Flea2 1280 x 960 cameras (see Fig. 5). So the experiment was undertaken with the same robot but with different cameras between the learning (with Marlin cameras) and replay (with Flea2 cameras) steps, exploited in a BiCam configuration. Moreover the experiment was undertaken with path learning and replay, done on different time, thus, the environment has changed between the two steps.

The robot follows the trajectory shown in Fig. 4. The main issues of this experiment are, first, learn a safe path, computing the learning-mapping approach and second, replay the path autonomously using the replay-localization and the path tracker method. During the first step, the learning-mapping procedure was used to map 3D-points landmarks and build the submaps. New local maps are created when 80 landmarks are in the map. In each submap, the poses and their uncertainties are expressed in the associated lrf. All submaps were merged in a final global map, expressed in wrf using the refinement of the global map procedure (see Fig. 6). As a second step to test the replay-localization approach, the replay-robot must replay the path learned during the learning-mapping procedure. We can note that the initial position of the robot in the replay step is not exactly the same that the initial position in the learning step. The 3D global map and the global path are loaded on the robot. In this point, the replay-robot, has a good knowledge of the 3D map of the environment, thus, the robot starts with a huge uncertainty and applies the algorithm of replay-localization. The results obtained in this experiment\(^1\), confirm the system performance in the learning-mapping procedure as well as the accuracy of the replay-localization along the path (Fig. 7). We can see that the replay step works in spite of bad matchings on landmarks occluded or lost, due to the environments changes (Fig. 8).

\(^1\)A representative video of the experimental results can be seen at http://homepages.laas.fr/dmarquez/maplaas
Fig. 4. View of the learned and replayed trajectory in the experiment. The robot follows the trajectory described by the sequence 0-1-2-3-4-5.

Fig. 5. The Segway platform used in the experiment. Two independent stereo-vision benches. The first one made of two Marlin cameras (red cameras) with a baseline of 0.40 m. The second one made of two Flea2 cameras (black cameras) with a baseline of 0.30 m.

Fig. 6. Learning and Replay experiment: 2D and 3D plots of the global map obtained by joining 67 local submaps using our hierarchical/hybrid BiCam SLAM approach with refinement of the global map procedure; red: covariance ellipses of the features; black: the estimated robot trajectory.

Fig. 7. Learning and Replay experiment: robot trajectory estimate; black: the robot trajectory learned path; red: the robot trajectory replayed path; blue: the true robot trajectory (GPS).

B. The Convoy task

A second experiment using MORSE\(^2\) was undertaken. MORSE is a domain independent simulator, where virtual robots can interact with a 3D environment, using sensors and actuators that behave in the same way as their counterparts in the real world. The main issues of this experiment is to make a “Convoy” with two robots (leader and follower) where the leader applies the learning-mapping approach to build successive submaps, and to transmit online each submap once it is built, defining a path that the follower must stay on. The follower, waits for the first submap to start and applies the replay-localization approach to follow the trajectory, processing the information received (online) from the leader. Thus the leader-robot start to explore the area and build the first local map, this first submap is the origin of the world. New local maps are created when 80 landmarks are in the map, but before starting a new local map, the submap and the local trajectory are expressed in wrf and transmitted to the follower-robot. Immediately after, the follower-robot pose is the new relative transformation in the global graph. As results, 20 submaps were generated and sent from the leader to the follower. The follower was able to process each submap and apply the replay-localization approach to track and follow (online) the path defined by the leader (see Fig. 9).

The results obtained in this experiment, confirm the system performance in the convoy task (see Fig. 10).

VI. CONCLUSIONS

We have designed and tested a navigation system in order to implement convoy navigation by robots that must navigate in dangerous unknown routes. Only perception results have been exhibited, a general method for trajectory control has

\(^2\)http://morse.openrobots.org
been used to maintain the follower on the learnt trajectory. A mixed strategy combining path tracking with visual servoing will be studied in the future.

It has been shown [16] that visual SLAM methods based on non linear optimization (Incremental Sparse Bundle Adjustment, or SBA) converge better than EKF-SLAM or more generally, methods based on filtering. Here why do we use an EKF-SLAM method? First in this application, loop closure is not considered: the method must both build the map and the trajectory, but the evaluation criterion will be based on the capability for a robot to replay a learned trajectory, whatever the drift with respect to the exact localization in a global frame. Non linear optimization is only used off line in order to refine the map and the learned trajectory. Then for the convoy configuration, off line refinement is forbidden. Thus, in our system, EKF-SLAM has been selected has the best way to generate submap’s that could be communicated from the leader robot to the followers ones.

REFERENCES